NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A THREE DIMENSIONAL
NON-SINGULAR MODELLING OF RIGID
MANIPULATORS

by

Sadrettin Altinok

December 1987

Thesis Advisor D.L. Smith

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT APPROVED FOR PUBLIC RELEASE DISTRIBUTION IS UNLIMITED
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION NAVAL POSTGRADUATE SCHOOL 6b OFFICE SYMBOL (If applicable) 69	7a NAME OF MONITORING ORGANIZATION NAVAL POSTGRADUATE SCHOOL
6c ADDRESS (City, State, and ZIP Code) MONTEREY, CALIFORNIA 93943-5000	7b ADDRESS (City, State, and ZIP Code) MONTEREY, CALIFORNIA 93943-5100
8a NAME OF FUNDING / SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
A THREE DIMENSIONAL NON-SINGULAR MODELLING OF RIGID MANIPULATORS

12 PERSONAL AUTHOR(S)
SADRETTIN ALTINOK

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
MASTERS THESIS	FROM _____ TO _____	1987 DECEMBER	106

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	ROBOTICS, ROBOT SIMULATION, SINGULARITY, MODELLING DYNAMIC EQUATIONS OF MOTION, NON-SINGULAR

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

There are several common approaches used to obtain the kinematic and dynamic equations which describe the motion of robot manipulators. However, a problem arises when these conventional body oriented robot arm kinematic equations are used to simulate the manipulator motion. In this case, the jacobian matrix which relates the end effector motion to joint angle variations becomes singular when two succesive arm links align. When the robot arm passes through these singular points, the jacobian matrix is not invertible, and a result of this, the motion cannot be simulated. This thesis investigates how this situation can be avoided by using a Newton Euler approach to variable definition, and using the equations interpretted in a fixed reference frame.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL DAVID SMITH	22b TELEPHONE (Include Area Code) (408)646-3383 22c OFFICE SYMBOL 69Sm

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

☆ U.S. Government Printing Office 1986—606-24

UNCLASSIFIED

Approved for public release; distribution is unlimited.

A Three Dimensional Non-Singular Modelling of Rigid Manipulators

by

Sadrettin Altinok
1st. Lieutenant, Turkish Army
B.S., Turkish Army Academy,1981

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1987

ABSTRACT

There are several common approaches used to obtain the kinematic and dynamic equations which describe the motion of robot manipulators. However, a problem arises when these conventional body oriented robot arm kinematic equations are used to simulate the manipulator motion. In this case, the jocobian matrix which relates the end effector motion to joint angle variations becomes singular when two successive arm links align. When the robot arm passes through these singular points, the jacobian matrix is not invertible, and a result of this, the motion cannot be simulated. This thesis investigates how this situation can be avoided by using a Newton Euler approach to variable difinition, and using the equations interpretted in a fixed reference frame.

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF THE SYMBOLS AND ABBREVIATIONS

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
A	A	Sine Wave Input Torque Amplitude
AA	Aa	Acceleration of Point A
AB	Ab	Acceleration of Point B
AG1	Ag1	The Acceleration Vector of the Center of Gravity for Link 1
AG2	Ag2	Same as AG1 but for Link 2
AG3	Ag3	Same as AG1 but for Link 3
AOX	aox	Linear Acceleration of Joint Zero in the X Direction
AOY	aoy	Linear Acceleration of Joint Zero in the Y Direction
AOZ	aoz	Linear Acceleration of Joint Zero in the Z Direction

AX1	ax1	Linear Acceleration of Link One in the X Direction
AY1	ay1	Linear Acceleration of Link One in the Y Direction
AZ1	az1	Linear Acceleration of Link One in the Z Direction
AX2	ax2	Linear Acceleration of Link Two in the X Direction
AY2	ay2	Linear Acceleration of Link Two in the Y Direction
AZ2	az2	Linear Acceleration of Link Two in the Z Direction
AX3	ax3	Linear Acceleration of Link Three in the X Direction
AY3	ay3	Linear Acceleration of Link Three in the Y Direction

AZ3	az3	Linear Acceleration of Link Three in the Z Direction
BRATE1(3)	Brate1	Angular Velocity Vector in Body Fixed (rotating) Coordinate System for Link 1 in the x, y and z Direction Respectively
BRATE2(3)	Brate2	Same as Brate1 but for the Link 2
BRATE3(3)	Brate3	Same as Brate1 but for the Link 3
DEGRA		Conversion from Degrees to Radians
DRCANX(1)		Direction Cosine Angle
DRCANX(2)		in Degrees in Fixed
DRCANX(3)		Coordinate System from X Axis for Link 1-3 Respectively
DRCANY(1)		Direction Cosine Angle
DRCANY(2)		in Degrees in Fixed
DRCANY(3)		Coordinate System from Y Axis for Link 1-3 Respectively
DRCANZ(1)		Direction Cosine Angle
DRCANZ(2)		in Degrees in Fixed

DRCANZ(3)	Coordinate System from Z Axis for Link 1-3 Respectively
DRCRAX(1) DRCRAX(2) DRCRAX(3)	Direction Cosine Angle in Radians in Fixed Coordinate System from X Axis for Link 1-3 Respectively
DRCRAY(1) DRCRAY(2) DRCRAY(3)	Direction Cosine Angle in Radians in Fixed Coordinate System from Y Axis for Link 1-3 Respectively
DRCRAZ(1) DRCRAZ(2) DRCRAZ(3)	Direction Cosine Angle in Radians in Fixed Coordinate System from Z Axis for Link 1-3 Respectively
DRCSX(1) DRCSX(2) DRCSX(3)	The Argument of Direction Cosine Angle in the x Direction for Link 1-3 Respectively
DRCSY(1) DRCSY(2) DRCSY(3)	The Argument of Direction Cosine Angle in the y Direction for Link 1-3 Respectively

DRCSZ(1)		The Argument of Direction
DRCSZ(2)		Cosine Angle in the z
DRCSZ(3)		Direction for Link 1-3
		Respectively
DQ(27)	DQ	A 27*1 Column Matrix
		Obtained by Multiplying
		the MatA and MatB
FXO	Fxo	Computed Force in the X
		Direction at Joint O
FYO	Fyo	Computed Force in the Y
		Direction at Joint O
FZO	Fzo	Computed Force in the Z
		Direction at Joint O
FX1	Fx1	Computed Force in the X
		Direction at Joint 1
FY1	Fy1	Computed Force in the Y
		Direction at Joint 1
FZ1	Fz1	Computed Force in the Z
		Direction at Joint 1
FX2	Fx2	Computed Force in the X
		Direction at Joint 2
FY2	Fy2	Computed Force in the Y
		Direction at Joint 2
FZ2	Fz2	Computed Force in the Z
		Direction at Joint 2
G	g	Gravitional Constant

HDX(2)	HDx	The Time Rate of Change of Angular Momentum of a Two Elements Composite Body in the X Direction
HDY(2)	HDy	Same as HDX but in the Y Direction
HDZ(2)	HDz	Same as HDX but in the Z Direction
I		Counter
IA		Row Dimension of Matrix A and Matrix B Used in LEQ2TF Subroutine
IER		Error Parameter Used in Subroutine LEQT2F
IDGT		Accuracy Test Used in Subroutine LEQT2F for Iterative Improvement
IXX(3,2)	Ixx	A 3*2 Matrix of Moment of Inertia for the Two Element Composite Body of Link 1-3 About X Axis
IYY(3,2)	Iyy	Same as IXX but About the Y Axis
IZZ(3,2)	Izz	Same as IXX but About the Z Axis

IXZ(3,2)	Ixz	A 3*2 Matrix of Products of Inertia for the Two Element Composite Body of Link 1-3 About the XZ Coordinate Axis
IXY(3,2)	Ixy	Same as IXZ but for the XY Axis
IYZ(3,2)	Iyz	Same as IXZ but for the YZ Axis
IXXT(3)		Total Moment of Inertia of Link 1-3 About X Axis
IYYT(3)		Same as IXXT but About the Y Axis
IZZT(3)		Same as IXXT but About the Z Axis
IXZT(3)		Same as IXXT but About the XZ Axis
IXYT(3)		Same as IXXT but About the XY Axis
IYZT(3)		Same as IXXT but About the YZ Axis
JXO	jxo	Location of Joint Zero in the X Direction
JYO	jyo	Location of Joint Zero in the Y Direction

JZO	jzo	Location of Joint Zero in the Z Direction
JX1	jx1	Location of Joint One in the X Direction
JY1	jy1	Location of Joint One in the Y Direction
JZ1	jz1	Location of Joint One in the Z Direction
JX2	jx2	Location of Joint Two in the X Direction
JY2	jy2	Location of Joint Two in the Y Direction
JZ2	jz2	Location of Joint Two in the Z Direction
L(3,2)	L(3,2)	A 3*2 Matrix that is the Distance from Center of Link to Center of Mass at Each Link End
LCOGX(3)	LCOGx	A 1*3 Location of Center of Gravity Vector for Link 1-3 in the X Direction
LCOGY(3)	LCOGy	Same as LCOGX but for the Y Direction
LCOGZ(3)	LCOGz	Same as LCOGX but for the Z Direction

M		Number of the Right Hand Side Used in LEQT2F
MASS(3,2)	Mass(3,2)	A 3*2 Matrix of Mass of Each Element that Make Up the Composite Body for Link 1-3
MASS1	M1	Total Mass of Link 1
MASS2	M2	Total Mass of Link 2
MASS3	M3	Total Mass of Link 3
MATA(27,27)	MatA	A 27*27 Matrix Consisting of Coefficients of the Unknown Variables
MATB(27)	MatB	A 27*1 Vector Consisting of the Coefficient of Known Variables on Input and an Output the Solution to the Linear Equations
MATC(27)	MatC	A 27*1 Column Matrix which Contains the Elements of the Known MatB in Simulation Process
MAT1R, MAT2R,	Mat1r Mat2r	Transformation Matricies from Fixed Coordinate

MAT3R	Mat3r	System to Body Fixed (Rotating) Coordinate System for Link 1 Link 2 and Link 3 Respectively
MAT1T, MAT2T, MAT3T	Mat1t Mat2t Mat3t	Transformation Matricies from Body Fixed Coordinate System to Yaw Pitch and Roll Angles Coordinate System for Link 1 Link 2 and Link 3 Respectively
MI		Results From Subroutine CPROD, I Component of Vector Cross Product
MJ		J Component of Vector Cross Product
MK		K Component of Vector Cross Product
MIAO,MJAO and MKAO		Cross Product Between WD1 and RB/G1 at Joint Zero Link One, in X, Y, Z Direction
MIBO,MJBO and MKBO		Cross Product Between W1 and RB/G1 at Joint Zero Link One, in X, Y, Z Direction

MICO,MJCO and MKCO	Cross Product Between W1 and MIBO, MJBO, MKBO at Joint Zero Link One, in X, Y, Z Direction
MIA1,MJA1 and MKA1	Cross Product Between WD1 and RA/G1 at Joint One Link One, in X, Y, Z Direction
MIB1,MJB1 and MKB1	Cross Product Between W1 and RA/G1 at Joint One Link One, in X, Y, Z Direction
MIC1,MJC1 and MKC1	Cross Product Between W1 and MIB1, MJB1, MKB1 at Joint One Link One, in X, Y, Z Direction
MIA2,MJA2 and MKA2	Cross Product Between WD2 and RB/G2 at Joint One Link Two, in X, Y, Z Direction
MIB2,MJB2 and MKB2	Cross Product Between W2 and RB/G1 at Joint One Link Two, in X, Y, Z Direction

MIC2,MJC2 and MKC2	Cross Product Between W2 and MIB2, MJB2, MKB2 at Joint One Link Two, in X, Y, Z Direction
MIA3,MJA3 and MKA3	Cross Product Between WD2 and RA/G2 at Joint Two Link Two, in X, Y, Z Direction
MIB3,MJB3 and MKB3	Cross Product Between W2 and RA/G2 at Joint Two Link Two, in X, Y, Z Direction
MIC3,MJC3 and MKC3	Cross Product Between W2 and MIB3, MJB3, MKB3 at Joint Two Link Two, in X, Y, Z Direction
MIA4,MJA4 and MKA4	Cross Product Between WD3 and RA/G3 at Joint Two Link Three, in X, Y, Z Direction
MIB4,MJB4 and MKB4	Cross Product Between W3 and RA/G3 at Joint Two Link Three, in X, Y, Z Direction

MIC4,MJC4 and MKC4		Cross Product Between W3 and MIB4, MJB4, MKB4 at Joint Two Link Three, in X, Y, Z Direction
N		Order of MATA and # of Rows in MATB
P		Phase Angle of Sine Wave
PTRY(1) PTRY(2) PTRY(3)		Pitch Angle in Radians for Link 1-3 Respectively
PTCANY(1) PTCANY(2) PTCANY(3)		Pitch Angle in Degrees for Link 1-3 Respectively
RADEG		Conversion from Radians to Degrees
RATE1(3)	Rate1	Angular Velocity Vector in Yaw Pitch and Roll Coordinate System for Link 1 in the x, y and z Direction Repectively
RATE2(3)	Rate2	Same as Rate1 but for the Link 2
RATE3(3)	Rate3	Same as Rate1 but for the Link 3

RX(3,2)	Rx(3,2)	A 3*2 Matrix Consisting of the Distance from the COG of the Link to Center of for Elements of Link 1-3 in the X Direction
RY(3,2)	RY(3,2)	Same as RX(3,2) but in the Y Direction
RZ(3,2)	Rz(3,2)	Same as RX(3,2) but in the Z Direction
RAG1(3)	ra/G1	A 1*3 Vector, Distance of Point A to COG for Link One, in X, Y, Z Direction
RBG1(3)	rb/G1	A 1*3 Vector, Distance of Point B to COG for Link One, X, Y, Z Direction
RAG2(3)	ra/G2	A 1*3 Vector, Distance of Point A to COG for Link Two, in X, Y, Z Direction
RBG2(3)	rb/G2	A 1*3 Vector, Distance of Point B to COG for Link Two, in X, Y, Z Direction

RBG3(3)	rb/G3	A 1*3 Vector, Distance of Point B to COG for Link Three, in X, Y, Z Direction
RLRZ(1)		Roll Angle in Radians for
RLRZ(2)		Link 1-3 Respectively
RLRZ(3)		
ROLANZ(1)		Roll Angle in Degrees for
ROLANZ(2)		Link 1-3 Respectively
ROLANZ(3)		
SUMHDX(3)	HDx	A 1*3 Vector, Sum of the HDX for the Two Elements of Link 1-3 in the X Direction
SUMHDY(3)	HDy	Same as HDX but in the Y Direction
SUMHDZ(3)	HDz	Same as HDX but in the Z Direction
TOX,TOY,	Tox,Toy	Input Torque at Joint O
TOZ	Toz	in X, Y, Z Direction
T1X,T1Y	T1x,T1y	Input Torque at Joint One
T1Z	T1z	in X, Y, Z Direction
T2X,T2Y	T2x,T2y	Input Torque at Joint Two
T2Z	T2z	in X, Y, Z Direction
TIPX,TIPY		Position of the End

TIPZ		Effector
VECTA0(3)		1*3 Vector Used in Subro-
VECTB0(3)		utine CPROD for Joint Zero
VECTA1(3)		1*3 Vector Used in Subro-
VECTB1(3)		utine CPROD for Joint One
VECTA2(3)		1*3 Vector Used in Subro-
VECTB2(3)		utine CPROD for Joint Two
VECTA(3)		1*3 Vector Used in Subro-
VECTB(3)		utine CPROD
W	w	Frequency of Sine Input
WG1,WG2	wg1,wg2	Weight of Links 1, 2, 3
WG3	wg3	
W1(3)	w1(3)	A 1*3 Vector of the Angular Velocity of Link 1 in x, y, and z Direction Respectively
W2(3)	w2(3)	Same as W1(3) but for the Link 2
W3(3)	w3(3)	Same as W1(3) but for the Link 3
WDX(3)	wdx(3)	Angular Acceleration of Link 1-3 in the X Direction

WDY(3)	wdy(3)	Angular Acceleration of Link 1-3 in the Y Direction
WDZ(3)	wdz(3)	Angular Acceleration of Link 1-3 in the Z Direction
WKAREA		Work Area for the LEQT2F
X1,X2,X3		Location of the COG for Link 1-3 in the X Direction
Y1,Y2,Y3		Location of the COG for Link 1-3 in the Y Direction
YWXR(1)		Yaw Angle in Radians for
YWXR(2)		Link 1-3 Respectively
YWXR(3)		
YAWANX(1)		Yaw Angle in Degrees for
YAWANX(2)		Link 1-3 Respectively
YAWANX(3)		
Z1,Z2,Z3		Location of the COG for Link 1-3 in the Z Direction

ACKNOWLEDGEMENTS

I would like to thank all those instructors that have contributed to the culmination of my educational experience at NPS. This thesis is a reflection of both, the knowledge they have imparted to me and the shaping of my intellect. In particular I would like to thank Professor Smith whose patience and help lead directly to the successful completion of my thesis. I would also like to thank Professor McGhee for his valuable ideas and suggestions.

I. INTRODUCTION

The study of robotics is a fairly new discipline. Although the roots of these studies and developments can be traced back to the 1940's, the first commercial computer controlled robot was not introduced until the late 1950's [Ref. 1]. Furthermore, as the theory developed, several common problemmatical methods have been widely accepted and used.

When robot motion is studied, it is usually divided into two parts: robot arm dynamics and robot arm kinematics. While the kinematics problem deals with the geometry of the arm links, the dynamics problem deals with the study of forced motion. The dynamics problem is further divided into two parts: the direct dynamics problem and the inverse dynamics problem. In the inverse dynamics problem, link variables such as acceleration and velocity are known and the forces and necessary joint torques for the desired motion are calculated. In the direct dynamics problem, the joint torques are known and the accelerations and velocities of each joint are calculated.

The kinematics problem is also divided into two parts: the direct kinematics problem and the inverse kinematics problem. The direct kinematics problem is, given a set of critical geometric joint and link variables for each of the

joint-link pairs and the joint angle vector, determine the position and orientation of the end effector of the manipulator. The Denavit-Hartenberg representation, which uses a homogeneous transformation matrix to describe the spatial relationships between two adjacent rigid mechanical links is the most common method used to study the direct kinematics problem [Ref. 2]. The inverse kinematics problem is, given a desired position and orientation of the end effector of the manipulator and a set of critical geometric joint and link parameters, determine the corresponding joint angle vector; i.e., find all of the joint angles of the robot arm so that the end effector can be positioned in the desired location.

A difficulty in the solution to the inverse kinematics problem arises when two successive links align [Ref. 3]. At these times the angle between two successive links becomes 0 or 180 degrees, and the Jacobian matrix which relates the end effector motion to the joint variable variations cannot be inverted. This means that motion cannot be simulated. Different approaches to this problem have been investigated. One method deals with the Newton-Euler approach with a moving coordinate system [Refs. 4, 5], another uses the Langrangian approach [Refs. 6, 7]. One method deals with a virtual work approach [Ref. 8]. Kane's dynamics equations have been used due to computational efficiency [Ref. 9].

However, none of these methods have been able to overcome this singularity problem [Ref. 3].

Several methods have been proposed to avoid the singular configuration. One method proposed to minimize the time near the singular points [Ref. 10], thereby reducing their effects. In another method, it was proposed to avoid these singular points by confining the motion [Ref. 11]. Other solutions deal with presenting equations that can translate the manipulator in the neighborhood of a singularity through identification of singular points beforehand [Refs. 12, 13, 14]. It has also been shown that the redundancy of robot manipulators is effective in dealing with the singularities [Refs. 15, 16, 17].

In this thesis the equations of motion are derived using the principles of Newtonian dynamics in terms of a globally fixed coordinate system to overcome the singularity problem. Each link is treated as a free body with forces and moments applied at the joints and free body analysis is used to derive the equations of motion. Although the equations are relatively long and the solution to the problem is computationally time consuming, it is shown that these equations do overcome the singularity problem. The direct dynamics and the inverse dynamics problem are both simulated.

II. THEORETICAL DEVELOPMENT

A. THEORY OF THE SOLUTION

To derive the non-singular equations of motion the Newton-Euler approach is used (Figure 1). Each link is treated as a free body with forces and moments applied to it, weight has been disregarded. The globally fixed X Y Z coordinate system is used for the equations. All links are assumed to be rigid, so the effects of flexibility are not considered. All of the distances and the directions of the forces and moments have been based on the fixed coordinate system rather than a local coordinate system which moves with the link [Refs. 4, 5]. The link masses, the initial link positions and the orientations are assumed to be known parameters. As a result of equation derivation in the fixed reference frame, the moment of inertia is allowed to change with respect to time and is calculated for each small integration interval. This is opposed to keeping inertia constant as used in the local coordinate formulations. But it is assumed that the moment of inertia is constant in each small integration interval. This last assumption effectively linearizes the equations of motion so that a non-singular matrix inversion can be used to solve the equation set.

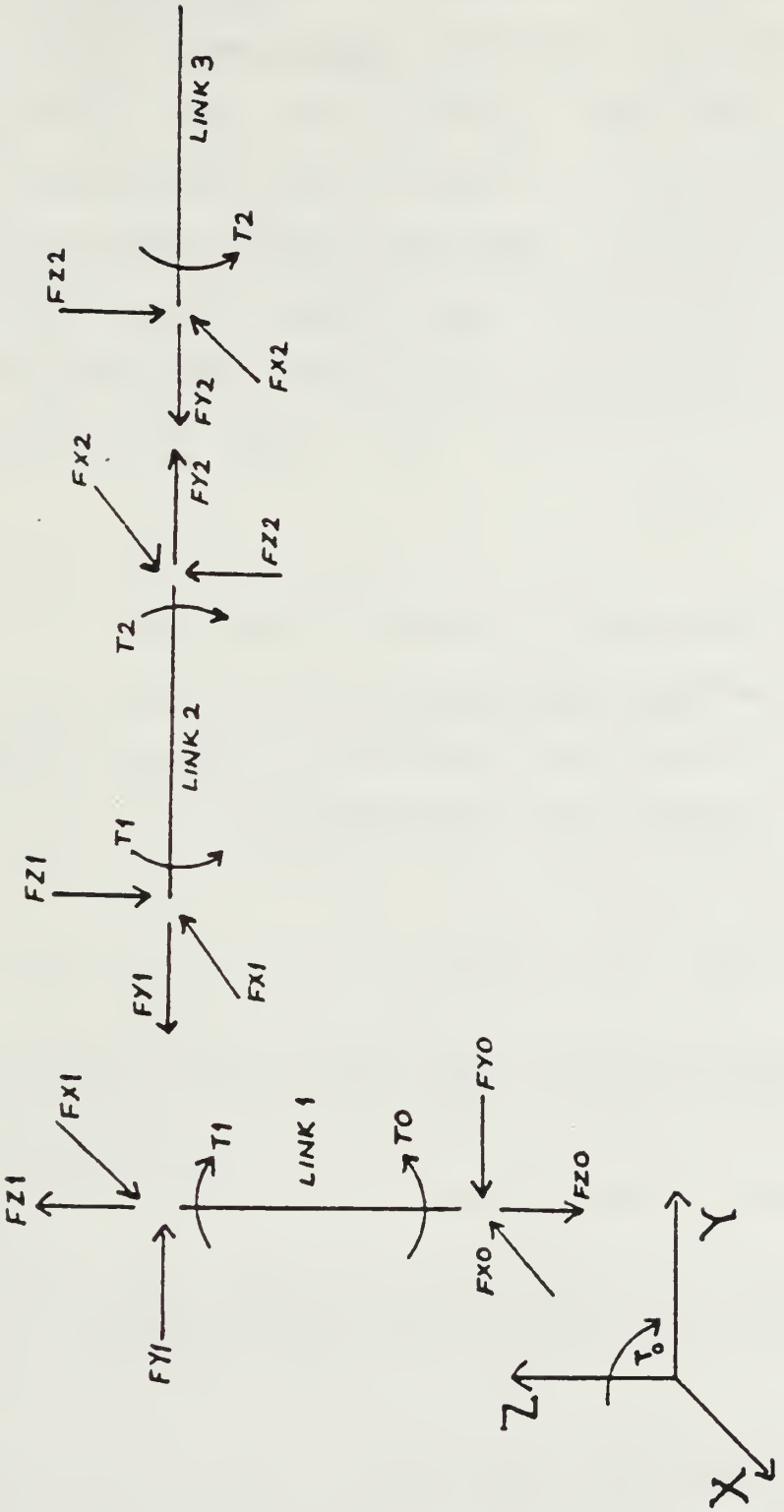


Figure 1. Free Body Diagram of a Three Link Manipulator

To calculate the moment of inertia in each integration interval, the link direction cosine angles with respect to the fixed coordinate system were used. Acceleration of joint zero was input as zero. For each link the three linear acceleration components, three angular acceleration components and forces at each joint were considered to be the unknown variables. Based on the Newtonian dynamics and the manipulator kinematics [Ref. 18], the equations were derived as follows:

B. DYNAMIC EQUATIONS OF MOTION OF LINK ONE

1. Sum of Forces Equations

In the free body analysis of link one (Figure 1) the sum of the forces in the x direction is:

$$\Sigma Fx = Fx1 - Fx0 = M1ax1 \qquad (1)$$

Similarly sum of the forces in the y direction is:

$$\Sigma Fy = Fy1 - Fy0 = M1ay1 \qquad (2)$$

and the sum of the forces in the z direction is:

$$\Sigma Fz = Fz1 - Fz0 - W1 = M1az1 \qquad (3)$$

2. Joint Equations

We begin by evaluating the joint equations at joint zero [Ref. 19, equation (8/4), pp. 423]. If the joint is sequested and analysis conducted at a point on link zero (subscript a) and another at a point on link one (subscript b) that is common to both, so when linked together they are equal. This results in two equations and the two unknowns wd1 and w1. As a result:

$$Aa = Ao$$

which is the acceleration at joint zero, and

$$Ab = A1 + (wd1 \times rb/G1) + w1 \times (w1 \times rb/G1)$$

which is the acceleration of point b on joint one. Here rb/G1 is the distance from point b to the center of gravity of link one, and A1 is the acceleration at the center of mass of link one or,

$$rb/G1 = (jx0-LC0Gx1)i + (jy0-LC0Gy1)j + (jz0-LC0Gz1)k$$
$$= rb/G1x + rb/G1y + rb/G1z$$

After equating Aa and Ab and having the known variables on the right side of the equation and unknown variables on the left side the following sets of equations result:

$$Ax1 + wdy1(rb/G1z) - wdz1(rb/G1y) = Aox - MIC0 \quad (4)$$

where MIC0 equals

$$= wy1wx1(rb/G1y) - (wy1)2(rb/G1x) - (wz1)2(rb/G1x) + wz1wx1(rb/G1z)$$

also

$$Ay1 + wdz1(rb/G1x) - wdx1(rb/G1z) = Aoy - MJC0 \quad (5)$$

where MJC0 equals

$$= wz1wy1(rb/G1z) - (wz1)2(rb/G1y) - (wx1)2(rb/G1y) + wx1wy1(rb/G1x)$$

and

$$Az1 + wdx1(rb/G1y) - wdy1(rb/G1x) = Aoz - MKC0 \quad (6)$$

MKC0 equals

$$= wx1wz1(rb/G1x) - (wz1)2(rb/G1z) - (wy1)2(rb/G1z) + wy1wz1(rb/G1y)$$

3. Sum of Moment Equations

Computing the sum of the moment equations about the center of gravity results in:

$$\Sigma M1 = (r0/G1 \times F0) + (r1/G1 \times F1) - T1 + T0$$

where the vector r0/G1 is the distance from joint zero to the center of gravity of link one and vector r1/G1 is the distance from joint one to the center of gravity of link one, in the x, y and z directions. Such that

$$r0/G1 = rj0 - rG1$$

and

$$r1/G1 = rj1 - rG1$$

so

$$rj0 - rG1 = (xj0 - xG1)i + (yj0 - yG1)j + (zj0 - zG1)k$$

and

$$rj1 - rG1 = (xj1 - xG1)i + (yj1 - yG1)j + (zj1 - zG1)k$$

In the x, y and z directions the sum of moment equations are:

$$\Sigma M1 \text{ in x direction} =$$
$$(yj0/G1)Fz0 + (zj0/G1)Fy0 + (yj1/G1)Fz1 - (zj1/G1)Fy1$$
$$- T1x + T0x \qquad (7a)$$

$$\Sigma M1 \text{ in y direction} =$$
$$(zj0/G1)Fx0 + (xj0/G1)Fz0 + (zj1/G1)Fx1 - (xj1/G1)Fz1$$
$$-T1y + T0y \qquad (8a)$$

$\Sigma M1$ in z direction=

$$(xj0/G1)Fy0 + (yj0/G1)Fx0 + (xj1/G1)Fy1 - (yj1/G1)Fx1 - T1z + T0z \quad (9a)$$

From [Ref.19, equation (57), pp. 227] the sum of the moments about a fixed point that does not move with the body is equal to the time rate of change of angular momentum of the system (H) about the fixed point, $\Sigma M = H$. In the present study we have let each link be a composite body of two elements. The angular momentum (H) for a composite body where the number of elements of the body is two, about the center of gravity of each link is $Hi = \Sigma [Ri \times (w \times Ri)]Mi$, where Ri is the distance from the center of gravity of each link to the appropriate element (1 or 2) in the x, y and z direction. So:

$$Hx = \Sigma [Ryi(wx(Ryi) - wy(Rxi)) - Rzi(wz(Rxi) - wx(Rzi))]Mi$$

$$Hx = [R^2y1(wx) - Ry1(Rx1)(wy) - Rzi(Rx1)(wz) + R^2z1(wx)]M1 + [R^2y2(wx) - Ry2(Rx2)(wy) - Rz2(Rx2)(wz) + (R^2z2)wx]M2$$

If $Ixx = Ry^2 + Rz^2\ dm$,

and $Ixy = RxRy\ dm$,

and $Ixz = RxRz\ dm$,

then:

$$Hx = [I1xx(wx) - I1xy(wy) - I1xz(wz)]M1 + [I2xx(wx) - I2xy(wy) - I2xz(wz)]M2$$

and

$$HDx = [I1xx(wdx) - I1xy(wdy) - I1xz(wdz)]M1 + [I2xx(wdx) - I2xy(wdy) - I2xz(wdz)]M2 \qquad (7b)$$

by assuming the moment of inertia changes with time but is constant for a given time interval.

By similar analysis it can be shown:

$$Hy = \Sigma [Rzi(wy(Rzi) - wz(Ryi)) - Rxi(wx(Ryi) - wy(Rxi) - wy(Rxi))]Mi$$

and if $Iyy = Rx^2 + Rz^2\ dm$,

and $Iyz = RyRz\ dm$,

and $Ixy = RxRy\ dm$,

then:

$$HDy = [I1yy(wdy) - I1yz(wdz) - I1yz(wdx)]M1 + [I2yy(wdy) - I2yz(wdz) - I2yx(wdx)]M2 \qquad (8b)$$

and

$$Hz = \Sigma [Rxi(wz(Rxi) - wx(Rzi)) - Ryi(wy(Rzi) - wz(Ryi))]Mi$$

if $Izz = Rx^2 + Ry^2\ dm$,

So $Hz = [I1zz(wz) - I1yz(wy) - I1zx(wx)]M1 + [I2zz(wz) - I2yz(wy) - I2zx(wx)]M2$

then

$$HDz = [I1zz(wdz) - I1yz(wdy) - I1zx(wdx)]M1 + [I2zz(wdz) - I2yz(wdy) - I2zx(wdx)]M2 \quad (9b)$$

Combining equations (7a) and (7b) and keeping known variables on the right side and unknown variables on the left side yields:

$$\Sigma M1x = (-yj0/G1)Fz0 + (zj0/G1)Fy0 + (yj1/G1)Fz1 - (zj1/G1)Fy1 - HDx = T1x - T0x \quad (7)$$

Combining equations (8a) and (8b) yields:

$$\Sigma M1y = (-zj0/G1)Fx0 + (xj0/G1)Fz0 + (zj1/G1)Fx1 - (xj1/G1)Fz1 - HDy = T1y - T0y \quad (8)$$

Combining equations (9a) and (9b) yields:

$$\Sigma M1z = -(xj0/G1)Fy0 + (yj0/G1)Fx0 + (xj1/G1)Fy1 - (yj1/G1)Fx1 - HDz = T1z - T0z \quad (9)$$

C. LINK TWO EQUATIONS

1. Sum of Forces Equations

From the free body diagram (Figure 1) it follows that

$$\Sigma Fx = Fx2 - Fx1 = M2ax2 \quad (10)$$

$$\Sigma Fy = Fy2 - Fy1 = M2ay2 \quad (11)$$

$$\Sigma Fz = Fz2 - Fz1 = M2az2 \quad (12)$$

2. Joint Equations

Analysis is conducted at joint one where similar equations are used as in joint zero with a point on link one (a) and one on link two (b). For point a the equation is

$$Aa = A1 + wd1 \times ra/G1 + w1 \times (w1 \times ra/G1)$$

ra/G1 is a vector whose distance is measured from point a to the center of gravity of link one in the x, y and z direction.

$$ra/G1 = (jx1 - LC0Gx1)i + (jy1 - LC0Gy1)j + (jz1 - LC0Gz1)k$$
$$= ra/G1x + ra/G1y + ra/G1z$$

For point b the equation is:

$$Ab = A2 + wd2 \times rb/G2 + w2 \times (w2 \times rb/G2)$$

where rb/G2 is a vector whose distance is measured from point b to the center of gravity of link two.

$$rb/G2 = (jx1 - LCOGx2)i + (jy1 - LCOGy2)j + (jz1 - LCOGz2)k$$
$$= rb/G2x + rb/G2y + rb/G2z$$

Equating Aa and Ab and setting knowns and unknowns on the respective sides of the equation results in:

$$Ax2 - Ax1 + wdy2(rb/G2z) - wdz2(rb/G2y) - wdy1(ra/G1z) + wdz1(ra/G1y) = MIC1 - M1C2 \quad (13)$$

$$MIC1 = wy1wx1(ra/G1y) - (wy1)2(ra/G1x) - (wz1)2(ra/G1x) + wz1wx1(ra/G1z)$$

$$MIC2 = wy2wx2(rb/G2y) - (wy2)2(rb/G2x) - (wz2)2(rb/G2x) + wz2wx2(rb/G2z)$$

$$Ay2 - Ay1 + wdz2(rb/G2x - wdx2(rb/G2z) - wdz1(ra/G1x) + wdx1(ra/G1z) = MJC1 - MJC2 \quad (14)$$

$$MJC1 = wz1wy1(ra/G1z) - (wz1)2(ra/G1y) - (wx1)2(ra/G1y) + wx1wy1(ra/G1x)$$

$$MJC2 = wz2wy2(rb/G2z) - (wz2)2(rb/G2y) - (wx2)2(rb/G2y) + wx2wy2(rb/G2x)$$

$$AZ2 - AZ1 + wdx2(rb/G2y) - wdy2(rb/G2x) - wdx1(ra/G1y) + wdy1(ra/G1x) = MKC1 - MKC2 \quad (15)$$

$$MKC1 = wx1wz1(ra/G1x) - (wx1)2(ra/G1z) - (wy1)2(ra/G1z) + wy1wz1(ra/G1y)$$

$$MKC2 = wx2wz2(rb/G2x) - (wx2)2(rb/G2z) - (wy2)2(rb/G2z) + wy2wz2(rb/G2y)$$

3. Sum of the Moment Equations

These equations have a similar development as that of link one:

$$\Sigma M2 = (rj1/G2) \times F1 + (rj2/G2) \times F2 + T1 - T2$$

where

$$rj1/G2 = (xj1 - xG2)i + (yj1 - yG2)j + (zj1 - zG2)k$$

$$rj2/G2 = (xj2 - xG2)i + (yj2 - yG2)j + (zj2 - zG2)k$$

$$\Sigma M2x = - (yj1 - yG2)Fz1 + (zj1 - zG2)Fy1 + (yj2 - yG2)Fz2 - (zj2 - zG2)Fy2 + T1x - T2x \quad (16a)$$

$$\Sigma M2y = -(zj1 - zG2)Fx1 + (xj1 - xG2)Fz1 + (zj2 - zG2)Fx2 - (xj2 - xG2)Fz2 + T1y - T2y \quad (17a)$$

$$\Sigma M2z = -(xj1 - xG2)Fy1 + (yj1 - yG2)Fx1 + (xj2 - xG2)Fy2 - (yj2 - yG2)Fx2 + T1z - T2z \quad (18a)$$

From angular momentum equation developed for link one, it can be shown for link two:

$$\Sigma M2x = HDx \quad (16b)$$

$$\Sigma M2y = HDy \quad (17b)$$

$$\Sigma M2z = HDz \quad (18b)$$

Combining equations (16a) and (16b) the following result:

$$-(yj1 - yG2)Fz1 + (zj1 - zG2)Fy1 + (yj2 - yG2)Fz2 - (zj2 - zG2)Fy2 - HDx = -T1x + T2x \quad (16)$$

Combining equations (17a) and (17b) yield the following result:

$$-(zj1 - zG2)Fx1 + (xj1 - xG2)Fz1 + (zj2 - zG2)Fx2 - (xj2 - xG2)Fz2 - HDy = -T1y + T2y \quad (17)$$

Combining equations (18a) and (18b) yield the following result:

$$- (xj1 - xG2)Fy1 + (yj1 - yG2)Fx1 + (xj2 - xG2)Fy2$$
$$- (yj2 - yG2)Fx2 - HDz = -T1z + T2z \qquad (18)$$

D. LINK THREE EQUATIONS

1. Sum of Forces Equations

Following similar logic from that previously developed:

$$\Sigma Fx = - Fx2 = M3ax3 \qquad (19)$$

$$\Sigma Fy = - Fy2 = M3ay3 \qquad (20)$$

$$\Sigma Fz = - Fz2 - W3 = M3az3 \qquad (21)$$

2. Joint Equations

With point a on link two and point b on link three one gets for joint equations at joint two:

$$Aa = A2 + (wd2 \times ra/G2) + w2 \times (w2 \times ra/G2)$$

where ra/G2 is a vector whose distance is measured from point a to center of gravity of link two in the x, y and z direction.

$$ra/G2 = (jx2 - LCOGx2)i + (jy2 - LCOGy2)j + (jz2 - LCOGz2)k$$
$$= ra/G2x + ra/G2y + ra/G2z$$

For point b

$$Ab = A3 + wd3 \times rb/G3 + w3 \times (w3 \times rb/G3)$$

where rb/G3 is a vector whose distance is measured from point b to center of gravity of link three in the x, y and z direction.

$$rb/G3 = (jx2 - LCOGx3)i + (jy2 - LCOGy3)j + (jz2 - LCOGz3)k$$
$$= rb/G3x + rb/G3y + rb/G3z$$

Equating Aa and Ab and setting knowns and unknowns on the respective sides of the equation results in:

$$Ax3 - Ax2 + wdy3(rb/G3z) - wdz3(rb/G3y) - wdy2(ra/G2z) + wdz2(ra/G2y) = MIC3 - MIC4 \quad (22)$$

$$MIC3 = wy2wx2(ra/G2y) - (wy2)2(ra/G2x) - (wz2)2(ra/G2x) + wz2wx2(ra/G2z)$$

$$MIC4 = wy3wx3(rb/G3y) - (wy3)2(rb/G3x) - (wz3)2(rb/G3x + wz3wx3(rb/G3z)$$

$$Ay3 - Ay2 + wdz3(rb/G3x) - wdx3(rb/G3z) - wdz2(ra/G2x) + wdx2(ra/G2z) = MJC3 - MJC4 \quad (23)$$

$$MJC3 = wz2wy2(ra/G2z) - (wz2)2(ra/G2y) - (wx2)2(ra/G2y) + wx2wy2(ra/G2x)$$

$$MJC4 = wz3wy3(rb/G3z) - w2z3(rb/G3y) - w2x3(rb/G3y) + wx3wy3(rb/G3x)$$

$$AZ3 - AZ2 + wdx3(rb/G3y) - wdy3(rb/G3x) - wdx2(ra/G2y) + wdy2(ra/G2x) = MKC3 - MKC4 \quad (24)$$

$$MKC3 = wx2wz2(ra/G2x) - (wx2)2(ra/G2z) - (wy2)2(ra/G2z) + wx2wy2(ra/G2y)$$

$$MKC4 = wx3wz3(rb/G3x) - (wx3)2(rb/G3z) - (wy3)2(rb/G3z) + wy3wz3(rb/G3y)$$

3. Sum of Moment Equations

As in the development of the equations for link one:

$$\Sigma M3 = (rj2/G3) \times F2 + T2$$

where

$$rj2/G3 = (xj2 - xG3)i + (yj2 - yG3)j + (zj2 - zG3)k$$
$$= xj2/G3 + yj2/G3 + zj2/G3$$

$$\Sigma M3x = (-yj2/G3)Fz2 + (zj2/G3)Fy2 + T2x \quad (25a)$$

$$\Sigma M3y = (-zj2/G3)Fx2 + (xj2/G3)Fz2 + T2y \quad (26a)$$

$$\Sigma M3z = (-xj2/G3)Fy2 + (yj2/G3)Fx2 + T2z \quad (27a)$$

From the angular momentum theory:

$$\Sigma M3x = HDx \quad (25b)$$

$$\Sigma M3y = HDy \quad (26b)$$

$$\Sigma M3z = HDz \quad (27b)$$

Combining equations (25a) and (25b) the following results:

$$(-yj2/G3)Fz2 + (zj2/G3)Fy2 - HDx = - T2x \quad (25)$$

Combining equations (26a) and (26b) the following results:

$$(-zj2/G3)Fx2 + (xj2/G3)Fz2 - HDy = - T2y \quad (26)$$

Combining equations (27a) and (27b) the following results:

$$(-xj2/G3)Fy2 + (yj2/G3)Fx2 - HDz = - T2z \quad (27)$$

III. COMPUTATIONAL APPROACH

A. PROGRAM MATRICIES

The Dynamic Simulation Language (DSL) was used to simulate the motion. This computer code was compiled on an IBM 3033 computer by using the FORTVS compiler and all of the calculations have been done in double precision. The entire simulation process is shown in Figure 2 and is discussed below.

The principle program matrix, Matrix A (MATA, 27*27), was created from the coefficients of the unknown variables in equations 1 to 27. In the simulation of the direct dynamics problem, a corresponding 27*1 Matrix B (MatB) was generated from all known variables, also from equations 1 to 27. A subroutine CPROD was used to perform all the cross product terms required in the main program. The resulting equations are shown in Figure 3, in the final matrix form. During a simulation time step, the link inertias, the link velocities and the link positions were all assumed constant. IMSL subroutine LEQT2F was called to invert the matrix A and get the generalized solution x from Ax = B. This subroutine uses Gaussian elimination with iterative improvement to get a high accuracy solution to the problem. The output from LEQT2F then returns as MATB, which contains the solution to the equations. The outputs were used by DSL to integrate

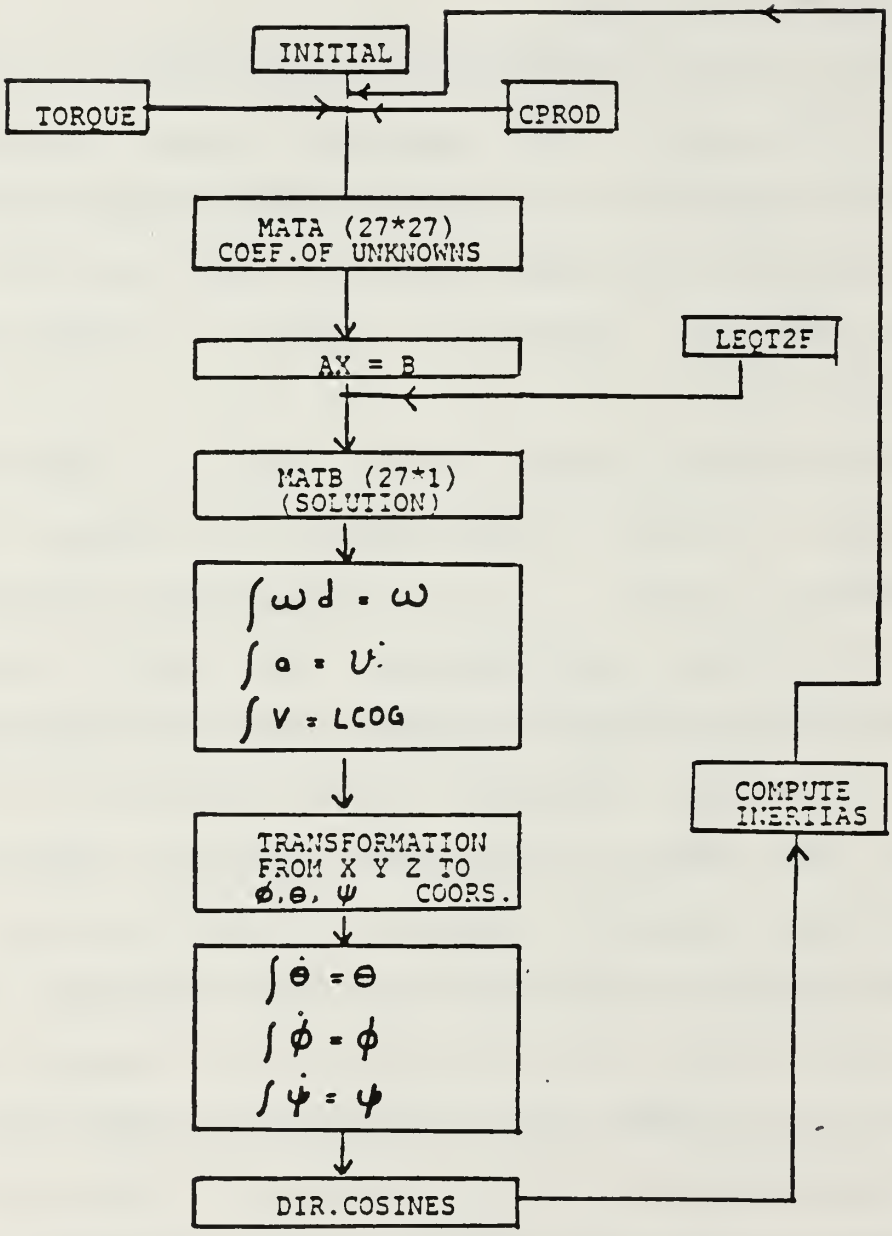


Figure 2. Computer Simulation Flow Chart

Figure 3. Matrix Entries

the linear and angular accelerations of each link to get the linear and angular velocities respectively. The linear velocities for each link were next integrated to get the linear displacements of center of gravity of each link. Although the linear velocities in the fixed reference frame can be integrated to get the linear displacements, this idea is not true for the angular displacements [Refs. 20, 21].

To get the angular displacements, a set of transformation matrices must be used on the velocities, then the motion can be integrated. That is, the angular velocities of each link in the fixed reference system must first be converted into equilavences in a body fixed coordinate system then into body Euler rates and Euler angles to define the motion unambigously. In this thesis, the body coordinate velocities are called Brate1, Brate2 and Brate3 for the link one, link two and link three respectively. To convert these velocities into the Euler rates, another transformation matrix is used. That is, the transformation matrix is multiplied by body rates to get the Euler angle rates for each link. These later rates are defined as the Yaw rate (about the x axis), the Pitch rate (about the y axis) and the Roll rate (about z axis). These rates are called as Rate1, Rate2 and Rate3 for link one, link two and link three. After the transformation of velocities to the Euler rates, they can be directly integrated to get the Euler angles. In this thesis, these

angles are called the Yaw, pitch and Roll angles about the x y z axis respectively [Refs. 2, 20].

This convention is very important and should not be mixed with another set of Euler angles described differently in the literature [Refs. 3, 20]. In addition to that, the order of the rotation must be decided beforehand. This is true because the orientation of objects is different when they are rotated in a different order, i.e., first the rotation about x axis, then a rotation about the y axis, finally a rotation about the z axis will produce a different orientation in space than the one which was defined and used in this thesis (z, y, then x). The transformation matrices used here are valid as long as the assumed order of the rotation is retained.

In the literature, a quite different set of angles is used to describe the orientation [Refs. 2, 3]. While some of these angles define the orientation with respect to a non-orthogonal coordinate system some others may define with respect to an orthogonal system. Euler angles define an independent set of coordinates system which are not orthogonal. Therefore, all three coordinates are independent from each other and velocities in this coordinate system can be directly integrated to get the relevant angles. They describe the unique orientation of the body in space. The orthogonal set of coordinate axes do not form an independent coordinate system. This is true

since the three axis have a certain relation with each other in any position, i.e., direction cosine angles have a unique relation in a fixed reference system and cannot be obtained by integrating any velocity in an orthogonal coordinate system. The velocities in an orthogonal coordinate system must thus be converted to a nonorthogonal coordinate system (e.g. Euler angle rates) prior to integration.

After Yaw, Pitch and Roll angles are calculated, it is possible to go back and express the orientation of the body with the direction cosines in an orthogonal coordinate system. The columns of the transformation matrix from one orthogonal set of axes to another describes the orientation of the new coordinate axis with respect to old coordinate system. So, a transformation matrix can be used to get the direction cosine angles. The direction cosines of each link are then used to calculate the moment of inertia of the links. The variation of a link inertia with respect to time was shown in Figure 4 as it was calculated during a simulation run. The derivation of the transformation matrices is shown in Appendix A.

B. CONSTRAINTS IN THE SIMULATION PROGRAM

In the development of the equations, thus far, each link has been treated such that it can move in space without any constraint. For most cases, however, degrees of freedom

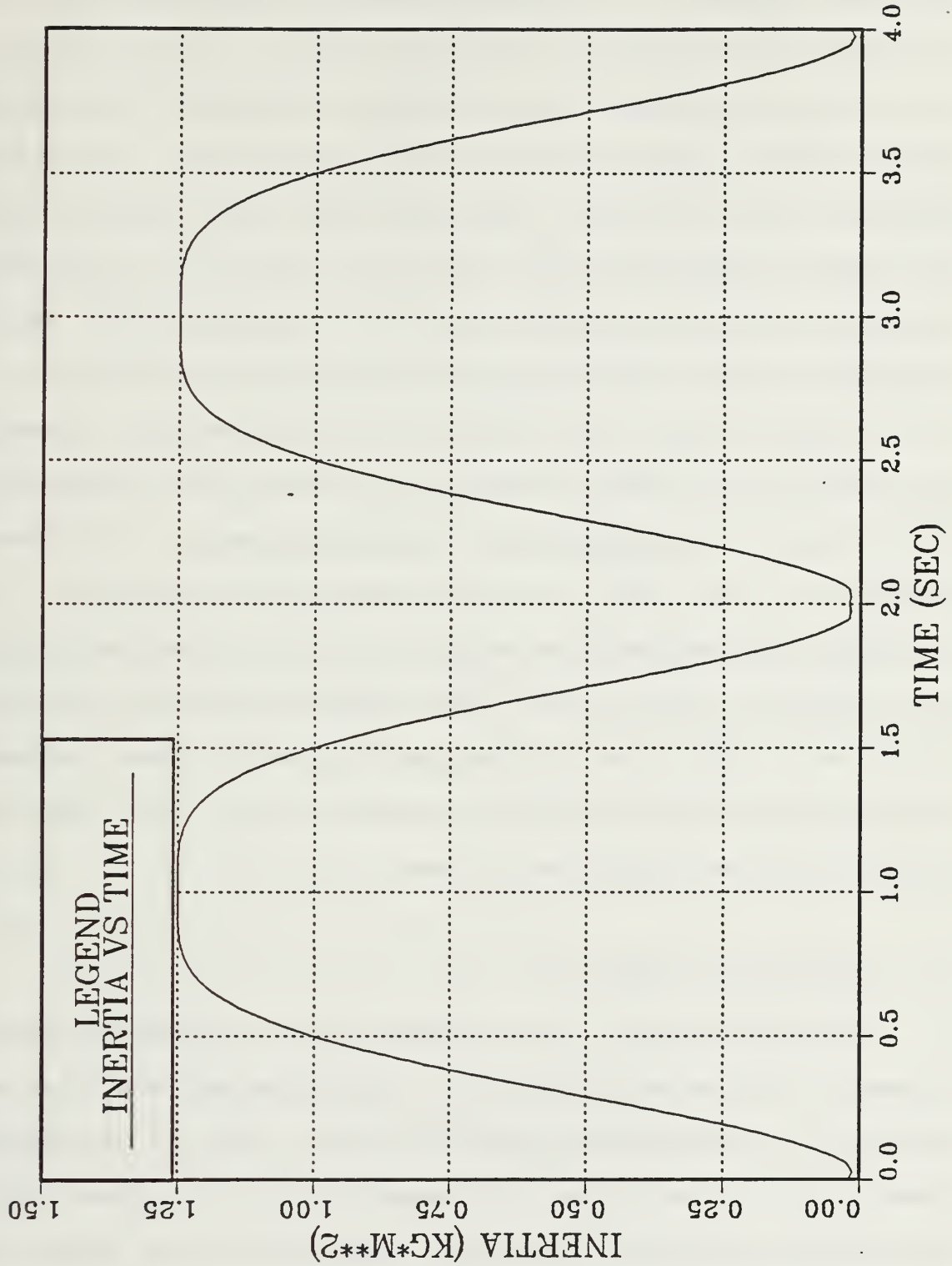


Figure 4. Moment of Inertia

of each link must be reduced so that the link can move only in the direction permitted by its joint.

In the simulation of the direct dynamics problem, the base rotation is transmitted to the second and third links for the three revolute joint arm which was studied. This was simulated by allowing the first link to rotate only about Z axis. At the same time, the rotational rates of the second and third links about the Z axis were made equal.

To make any of the simulation variables zero, meaning no variation in that direction, one zeroes the related rows and columns in MatA putting 1 on the diagonal. At the same time, if the same row in MatB is made zero, the corresponding mathematical expression for this equation will be in the form of 1 * X = 0, and a result of this, X will be equal zero. This idea can also be applied to MatA and MatB to make two variables equal so that X1 - X2 = 0. Thus, the above motion was simulated by constraint.

C. PROGRAM VALIDATION

The validation of the inverse dynamics problem has been conducted in several cases. In this approach the idea was to choose an angular acceleration such that at a certain time, two of the three links would align. In other words, the links would be in a singular position at this time, and if the simulation procedure worked, the singularity problem would have been avoided.

The validation procedure is shown in Figure 5. For this procedure link two angle was chosen as $\theta = (Pi/2) * \sin(pi/2) * Time$. This time dependent function has a period of 4 sec and an amplitude of 90 degrees. The second derivative of this function is $\theta = -((pi**3)/8) * \sin(pi/2) * Time$ and corresponds to the angular acceleration of the link. This value was input as the theoretical angular acceleration in the simulation program, and corresponding linear accelerations and forces at each joint were calculated. The other two links were forced to have zero rotational velocity throughout the simulation.

To apply a corresponding torque at the joint, MaTA and MatB were multiplied and a right hand side matrix DQ (27*1) was obtained (MATA * MATB = DQ). This matrix DQ (27*1) was used to solve the simulation equations in the form of AX = DQ. The vector X (that is, theta) was fedback in the loop and the theoretical and the calculated values of theta were compared.

The above discussion has been implemented in three different initials configuration as shown in Figure 6. To force the arm links to the various singular points, several different plane motions were simulated. For each configuration, three different angular motions were input for link 2, or as can be seen from Figure 6, for each configuration, one angular velocity caused a spinning motion of the link about the axis with which it was initially

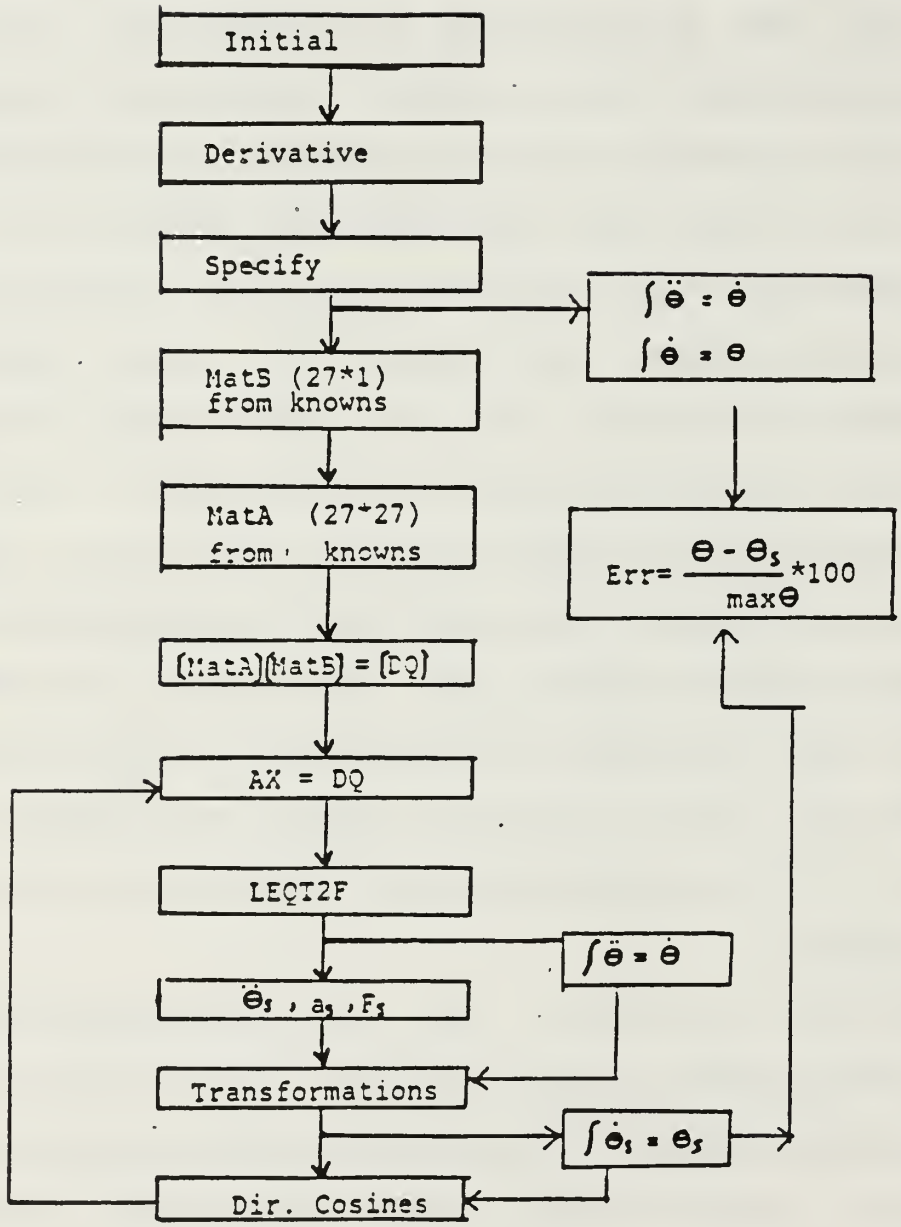


Figure 5. Validation Procedure Flow Chart

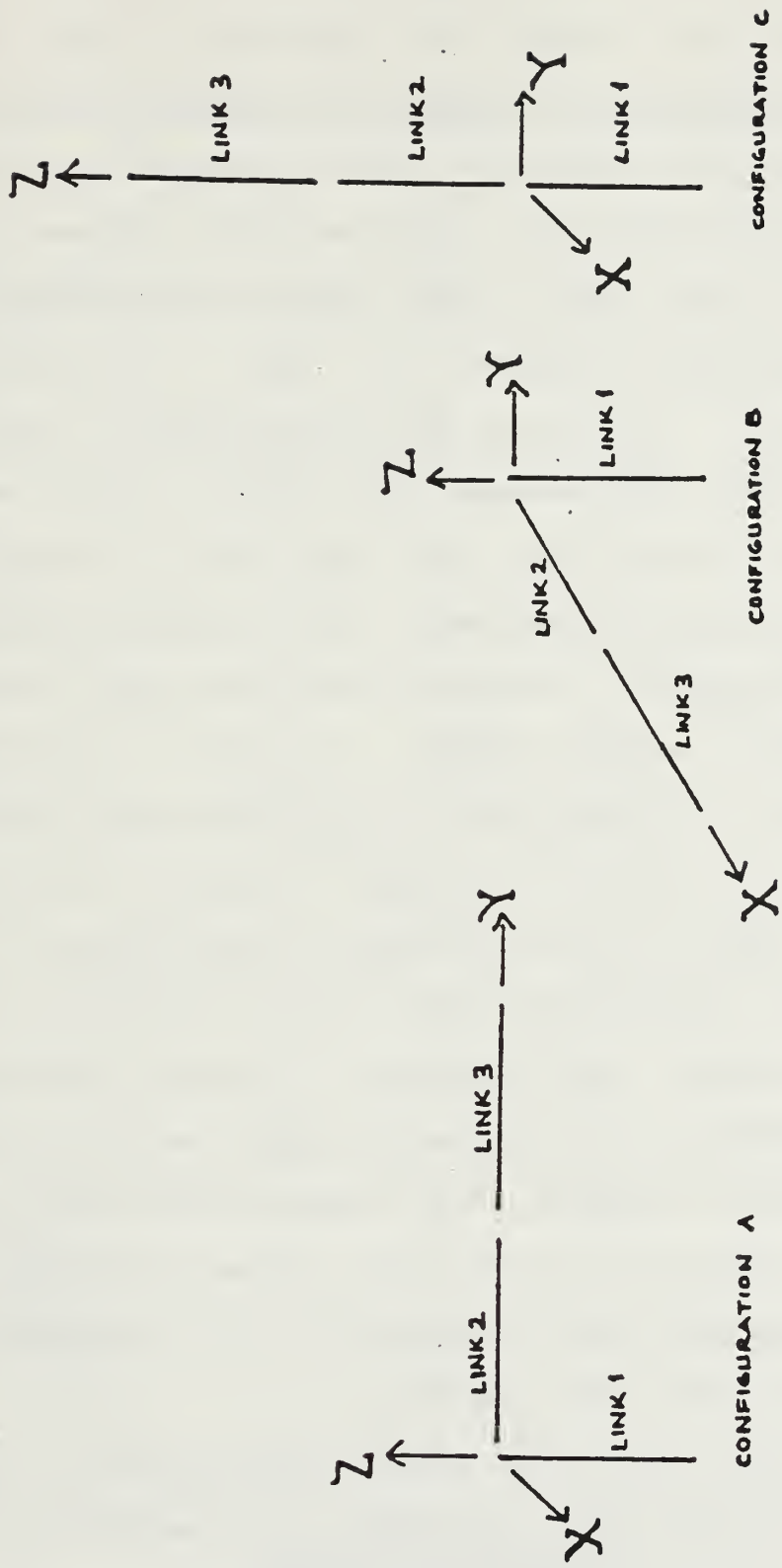


Figure 6. Initial Orientation of Links for Validation

aligned, while the other two produced a plane motion according to direction of the applied angular motion. The angles between two successive links were measured for each motion. Figure 7 shows the angle variation between link 1 and link 2, and link 2 and link 3 corresponding to an angular acceleration applied in the X direction for configuration A. As can be seen from Figures 7 and 8, two successive links pass through the singular points every 2 seconds, i.e., they align and the angle between links becomes either 0 or 180 degrees. (The singular points are marked on the graph). Figure 8 shows the angle variations for an angular motion applied in the Z direction for configuration A. In this case, it is obvious that the angle between link 1 and link 2 is always constant (90 degrees). The second graph on Figure 8 shows the angle between link 2 and link 3 now, singularity occurs on the Z motion, with the singularities marked as in Figure 8. Figure 7 and Figure 8 are representative of the data obtained in the validation procedure which analyzed nine possible motions of link 2 leading the singularity. This data showed that singularity in these directions could be overcome, and a solution to the problem exists using this approach.

For each run, the error between the theoretical and the simulated value of Theta was computed. Figure 9 shows the percent error for the X motion for configuration A (Figure 7 Data). The trend of the error is representative of every

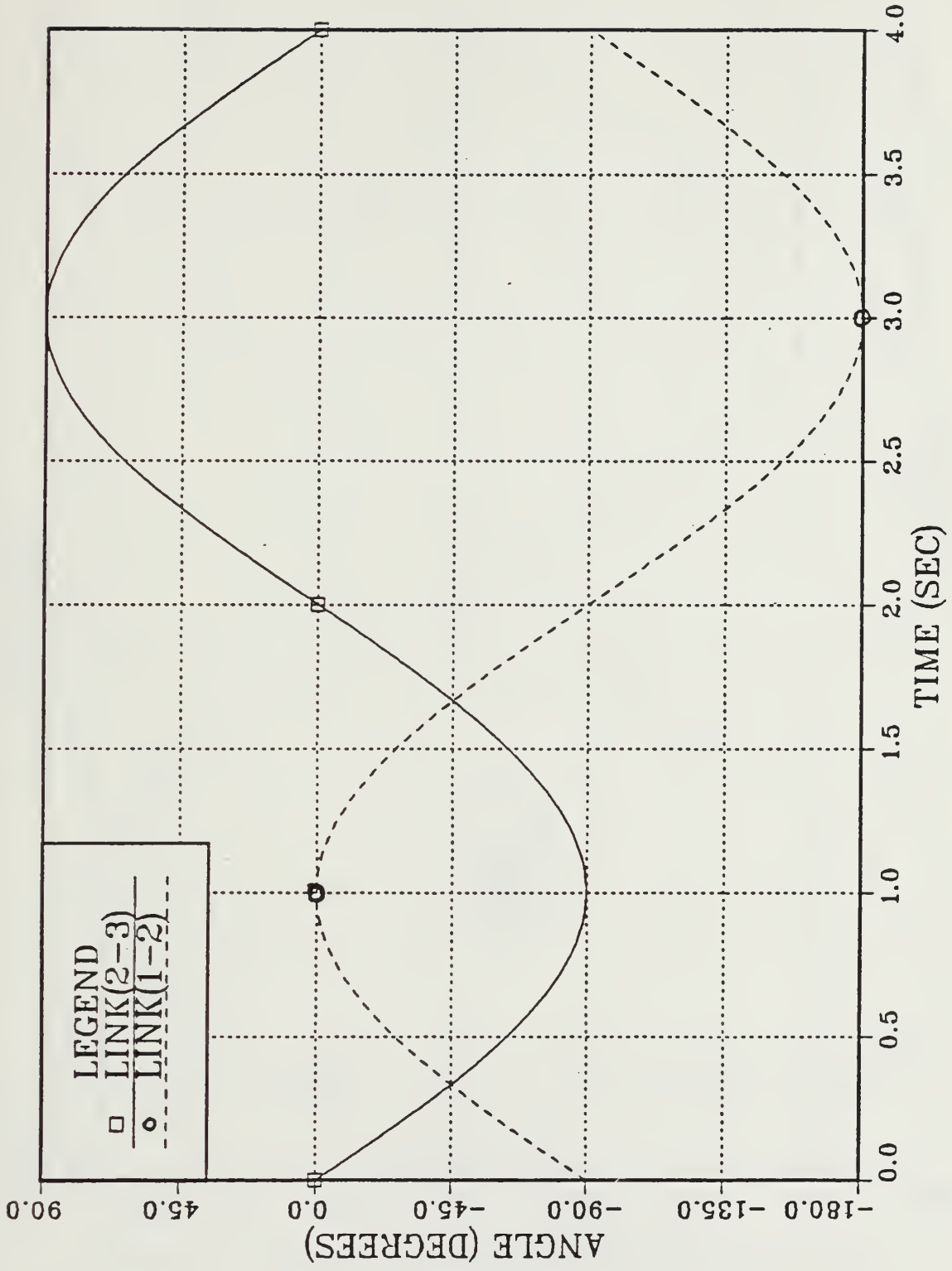


Figure 7. Configuration A, Link 2 Wx Motion

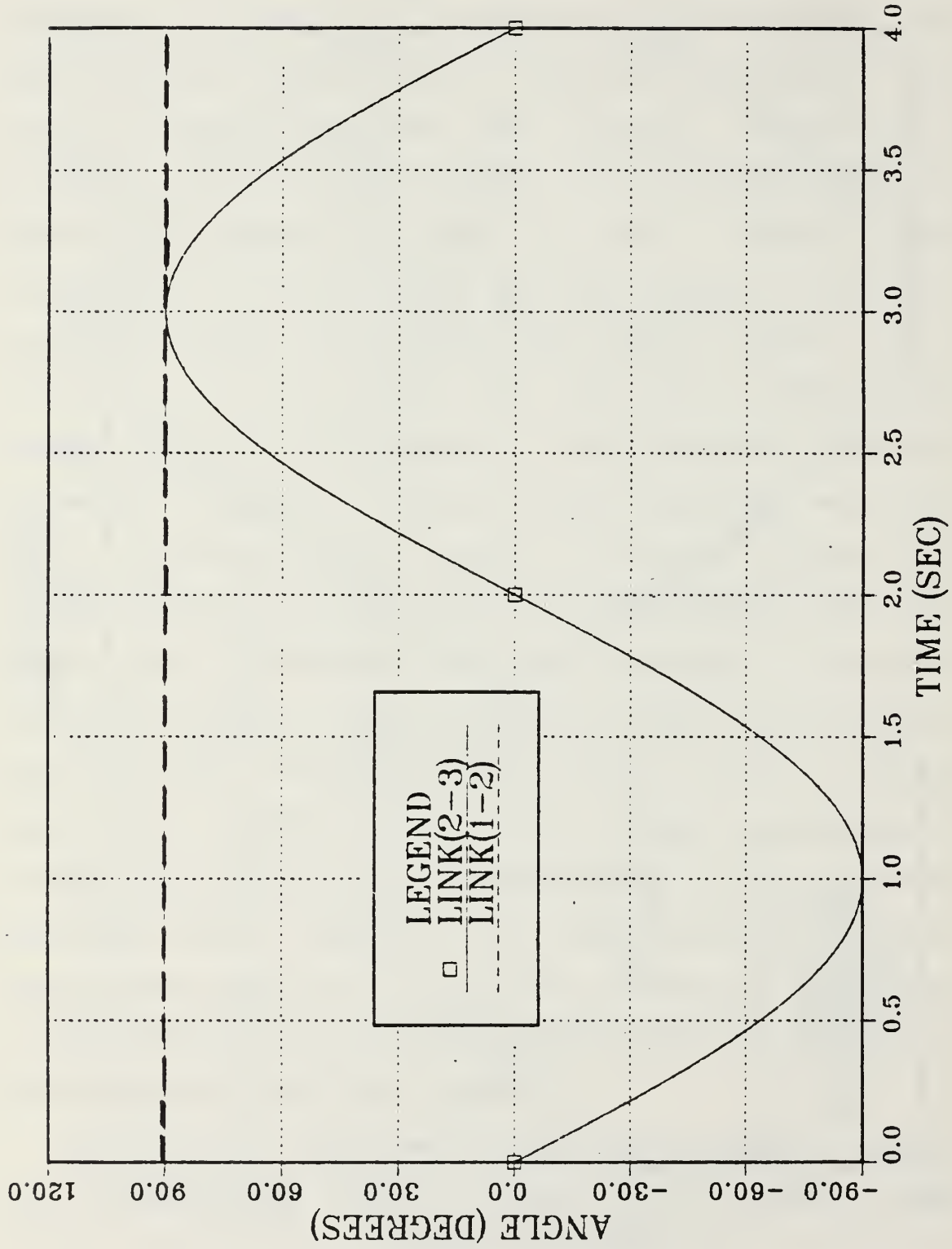


Figure 8. Configuration A, Link 2 W_z Motion

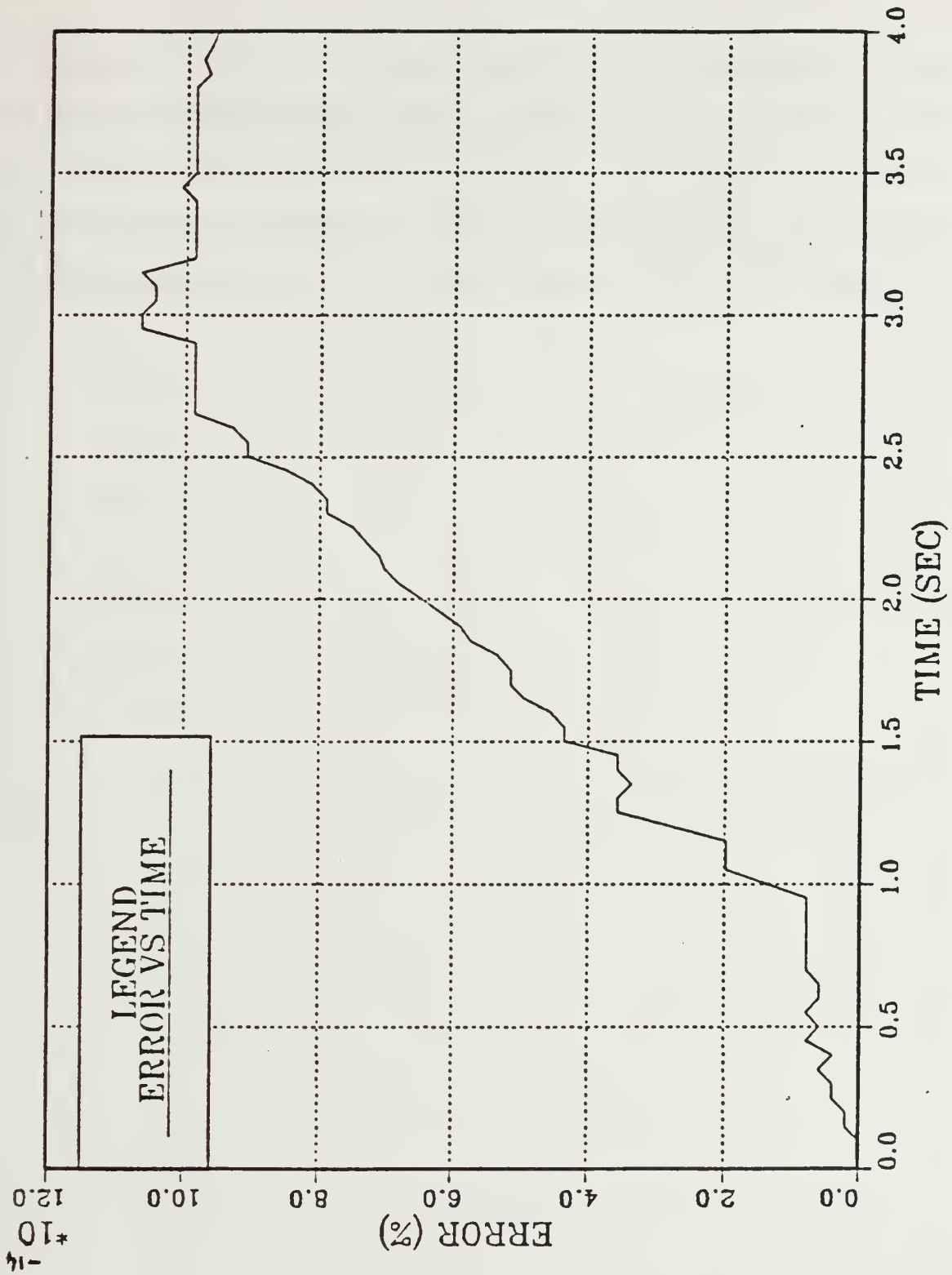


Figure 9. Percent Error Between Theoretical and Simulated Angles

case investigated. The figure shows that, due to nature of the numerical integration, the error slightly accumlates during the simulation, but still has very small value. This proves that the direct dynamics problem can be solved very accurately by Newton-Euler approach in a fixed coordinate system.

IV. RESULTS AND RECOMMENDATIONS

1. A dynamic model of a three link, rigid revolute joint manipulator has been developed in this thesis, as a general computer program package.
2. Several runs for different initial configurations were simulated and the singularity problem was investigated. Theoretical and calculated values of angular positions were compared. It was proved that the singularity problem could be overcome by using a Newton-Euler approach in a fixed coordinate system.
3. The following recommendations are provided:
   a. Enhance the code and make it more interactive. That is, let the user specify the constraints he wants to apply on each link by answering interactive questions before the actual simulation run starts. Thus, the motion can be simulated with different constraints without going into the code and changing the relevant parameters.
   b. Adapt the code for use in a microcomputer. Add a subroutine in the program to invert the matrix A. Thus, the code will be more independent from outside routines and more adaptable to other computer systems.

c. Validation of the approach via actual experimental tests in crucial. This will establish a way of developing accurate constants for subsequent controller design and provide a basis for compensation of gravity effects. Determining these constants for the code will make the simulation program more concrete and will provide more physical insight.

d. Finally, develop a controller for a manipulator which makes use of the present algorithm for validation and design.

APPENDIX A

DERIVATION OF THE TRANSFORMATION MATRIX FROM EARTH FIXED COORDINATE SYSTEM TO BODY FIXED COORDINATE SYSTEM

The angular velocity terms obtained by integration of the angular acceleration terms are with respect to an Earth fixed coordinate system. To define the Euler angles which are called Yaw Pitch and Roll angles in this thesis, we have to establish an appropriate body fixed coordinate system. Thus, U, V and W is a right hand coordinate system [Ref. 20] with its origin fixed at the center of gravity of a link. The U, V, W coordinate system is initially oriented such that the angles between two coordinate system axes are simultaneously reduced to zero, i.e., i, j, k, axis are parellel to the I, J, K respectively.

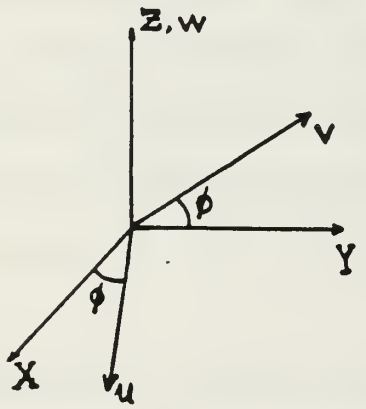
If a rotation from X Y Z coordinate system to the U V W coordinate system is accomplished by first rotation about K axis (roll), then about J axis (pitch) and finally about I axis (yaw), it follows that for any arbitrary point in the X Y Z coordinate system, the corresponding coordinates in the U V W system are;

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} \text{MatR} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

where MatR is a 3*3 matrix.

To get the transformation matrix, we need to examine each rotation separately.

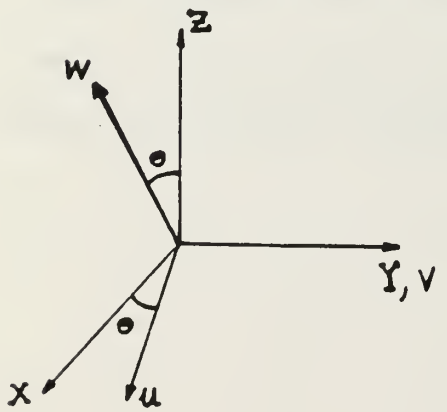
Rotation about the Z axis;



$$u = Xc\phi + Ys\phi$$
$$V = -Xs\phi + Yc\phi$$
$$W = Z$$

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} c\phi & s\phi & 0 \\ -s\phi & c\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

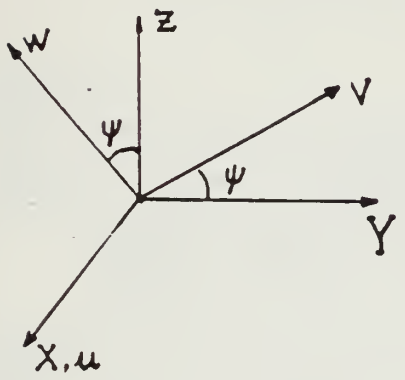
Rotation about the Y axis;



$$u = Xc\theta - Zs\theta$$
$$V = Y$$
$$W = Xs\theta + Zc\theta$$

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Rotation about the X axis;



$$u = X$$
$$V = Y C\psi + Z S\psi$$
$$W = -Y S\psi + Z C\psi$$

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\psi & s\psi \\ 0 & -s\psi & c\psi \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

By multiplying three rotation matrix together;

$$MATR = \begin{bmatrix} c\theta c\phi & c\theta s\phi & -s\theta \\ s\psi s\theta c\phi - c\psi s\phi & s\psi s\theta s\phi + c\psi c\phi & s\psi c\theta \\ c\psi s\theta c\phi + s\psi s\phi & c\psi s\theta s\phi - s\psi c\phi & c\psi c\theta \end{bmatrix}$$

where C = COS

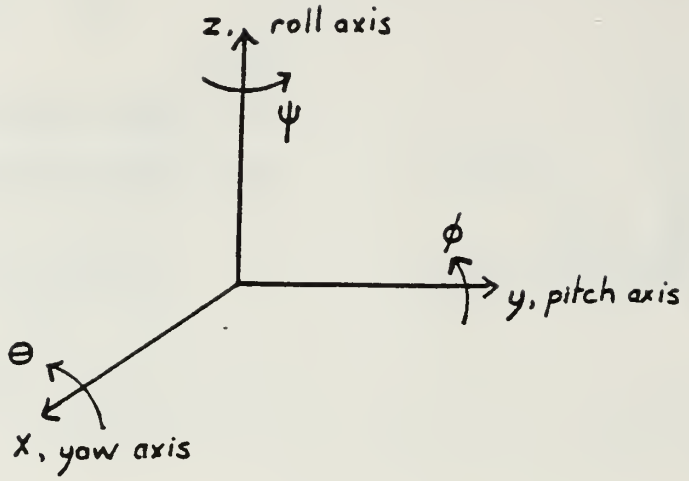
S = Sin

T = Tan

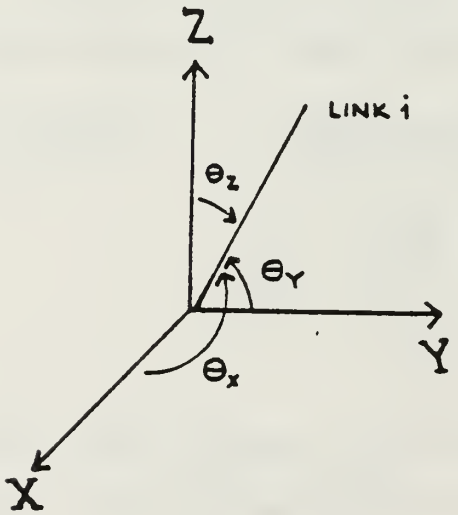
The transformation matrix from body fixed to Euler coordinate system is obtained as below [Ref. 20].

$$\begin{bmatrix} 0 & c\psi & -s\psi \\ 1 & T\theta s\psi & T\theta c\phi \\ 0 & Sec\theta s\psi & Sec\theta c\psi \end{bmatrix}$$

The angles discussed above are shown in Figure 10.



(a)



(b)

Figure 10. Critical Angles

(a) Euler Angles, Body Coordinates

(b) Direction Cosines, Global Coordinates

APPENDIX B

THREE DIMENSIONAL DIRECT DYNAMICS SIMULATION PROGRAM

```
TERMINAL
METHOD ADAMS
PRINT .05,DRCANX(1-3),DRCANY(1-3),DRCANZ(1-3),...
JX0,JY0,JZ0,JX1,JY1,JZ1,JX2,JY2,JZ2,...
LCOGX(1-3),LCOGY(1-3),LCOGZ(1-3)
CONTROL FINTIM =2.0, DELMAX =.1, DELPRT = .05
D     DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D     DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D     DIMENSION MAT1R(3,3),MAT2R(3,3),MAT3R(3,3)
D     DIMENSION MAT1T(3,3),MAT2T(3,3),MAT3T(3,3)
D     INTEGER IER,I,J,M,K,P,N,IA,IDGT,A
EXCLUDE IA,IDGT,IER,I,J,M,K,P,N,A
ARRAY MATB(27),LCOGX(3),LCOGY(3),LCOGZ(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY WDX(3),WDY(3),WDZ(3),W1(3),W2(3),W3(3)
ARRAY RATE1(3),RATE2(3),RATE3(3),BRATE1(3),BRATE2(3),BRATE3(3)
ARRAY RBG1(3),RAG1(3),RBG2(3),RAG2(3),RBG3(3)
ARRAY SUMHDX(3),SUMHDY(3),SUMHDZ(3),HDX(2),HDY(2),HDZ(2),WKAREA(850)
ARRAY IXXT(3),IYYT(3),IZZT(3),IXYT(3),IXZT(3),IYZT(3)
ARRAY YAWANX(3),PTCANY(3),ROLANZ(3)
ARRAY DRCANX(3),DRCANY(3),DRCANZ(3)
ARRAY DRCRAX(3),DRCRAY(3),DRCRAZ(3)
ARRAY DRCSX(3),DRCSY(3),DRCSZ(3)
D     DATA MATA/729 * 0.0D0/

INITIAL
*       INPUT PARAMETER CONSTANTS
            A = 5.0D0
            P = 0.0D0
            W = 2.0D0 * PI
            IDGT = 3
            G=0.0D0
            N=27
            M=1
            IA =27
*       INPUT JOINT LOCATIONS IN METERS
            JX0 = 0.0D0
            JY0 = 0.0D0
            JZ0 = 0.0D0
            JX1 = 0.0D0
            JY1 = 0.0D0
            JZ1 = 1.0D0
            JX2 = 0.0D0
            JY2 = 1.0D0
            JZ2 = 1.0D0
*       INPUT TORQUE CONSTANTS
            T0X = 0.0D0
            T0Y = 0.0D0
            T0Z = 0.0D0
            T1Y = 0.0D0
            T1Z = 0.0D0
            T2Y = 0.0D0
            T2Z = 0.0D0
*       INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS
```

```
*       FOR EACH LINK ENDS
            L(1,1) = 0.50D0
            L(1,2) = 0.50D0
            L(2,1) = 0.50D0
            L(2,2) = 0.50D0
            L(3,1) = 0.50D0
            L(3,2) = 0.50D0
*       INPUT MASS AT LINK ENDS IN KILOGRAMS
            MASS(1,1) = 2.5D0
            MASS(1,2) = 2.5D0
            MASS(2,1) = 2.5D0
            MASS(2,2) = 2.5D0
            MASS(3,1) = 2.5D0
            MASS(3,2) = 2.5D0
*       INPUT OMEGA AND OMEGA DOT, YAW, PITCH, AND ROLL ANGLES
            DO 30 I = 1,3
               W1(I)     = 0.0D0
               W2(I)     = 0.0D0
               W3(I)     = 0.0D0
               WDX(I)    = 0.0D0
               WDY(I)    = 0.0D0
               WDZ(I)    = 0.0D0
               YAWANX(I) = 0.0D0
               PTCANY(I) = 0.0D0
               ROLANZ(I) = 0.0D0
 30         CONTINUE
            YWRX1 = YAWANX(1) * DEGRA
            PTRY1 = PTCANY(1) * DEGRA
            RLRZ1 = ROLANZ(1) * DEGRA
            YWRX2 = YAWANX(2) * DEGRA
            PTRY2 = PTCANY(2) * DEGRA
            RLRZ2 = ROLANZ(2) * DEGRA
            YWRX3 = YAWANX(3) * DEGRA
            PTRY3 = PTCANY(3) * DEGRA
            RLRZ3 = ROLANZ(3) * DEGRA
*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
            LCOGX(1) = 0.0D0
            X1   =  LCOGX(1)
            LCOGY(1) = 0.0D0
            Y1   =  LCOGY(1)
            LCOGZ(1) = 0.5D0
            Z1   =  LCOGZ(1)
            LCOGX(2) = 0.0D0
            X2   =  LCOGX(2)
            LCOGY(2) = 0.5D0
            Y2   =  LCOGY(2)
            LCOGZ(2) = 1.0D0
            Z2   =  LCOGZ(2)
            LCOGX(3) = 0.0D0
            X3   =  LCOGX(3)
            LCOGY(3) = 1.5D0
            Y3   =  LCOGY(3)
            LCOGZ(3) = 1.0D0
            Z3   =  LCOGZ(3)
*       INPUT MASS OF EACH LINK IN KG AND COMPUTE WEIGHTS IN NEWTONS
            MASS1 = 5.0D0
            MASS2 = 5.0D0
            MASS3 = 5.0D0
            WG1 = MASS1*G
            WG2 = MASS2*G
            WG3 = MASS3*G
*       INPUT ACCELERATION OF JOINT ZERO
```

```
            A0X = 0.0D0
            A0Y = 0.0D0
            A0Z = 0.0D0
*       INITIALIZE MATRIX A AND B TO ZERO
            DO 40 I = 1,27
              DO 50 J = 1,27
                MATA(I,J) = 0.0D0
 50           CONTINUE
 40         CONTINUE
            DO 60 I = 1,27
              MATB(I) = 0.0D0
 60         CONTINUE
*       INITIALIZE THE TRANSFORMATION MATRICIES AND VELOCITIES
            DO 63 I = 1,3
              DO 64 J = 1,3
                RATE1(I)    =  0.0D0
                RATE2(I)    =  0.0D0
                RATE3(I)    =  0.0D0
                BRATE1(I)   =  0.0D0
                BRATE2(I)   =  0.0D0
                BRATE3(I)   =  0.0D0
                MAT1T (I,J) =  0.0D0
                MAT2T (I,J) =  0.0D0
                MAT3T (I,J) =  0.0D0
                MAT1R (I,J) =  0.0D0
                MAT2R (I,J) =  0.0D0
                MAT3R (I,J) =  0.0D0
 64           CONTINUE
 63       CONTINUE

DERIVATIVE
NOSORT
          CALL ERRSET (208,256,-1,1,1)
          LEVELQ = 0
          CALL UERSET(LEVELQ,LEVLDQ)
*       INITIALIZE MATRIX A AND B TO ZERO
            DO 70 I = 1,27
              DO 80 J = 1,27
                MATA(I,J) = 0.0D0
 80           CONTINUE
 70         CONTINUE
            DO 90 I = 1,27
              MATB(I) = 0.0D0
 90         CONTINUE

*       INPUT JOINT EQUATIONS
*       JOINT ZERO  EQUATIONS
*       AB = AG1 + (WD1 X RB/G1) + W1 X (W1 X RB/G1)
              VECTA0(1) = WDX(1)
              VECTA0(2) = WDY(1)
              VECTA0(3) = WDZ(1)
              RBG1(1) = JX0 - LCOGX(1)
              RBG1(2) = JY0 - LCOGY(1)
              RBG1(3) = JZ0 - LCOGZ(1)
            CALL CPROD(VECTA0,RBG1,MIA0,MJA0,MKA0)
              VECTA0(1) = W1(1)
              VECTA0(2) = W1(2)
              VECTA0(3) = W1(3)
            CALL CPROD(VECTA0,RBG1,MIB0,MJB0,MKB0)
```

```
                VECTB0(1) = MIB0
                VECTB0(2) = MJB0
                VECTB0(3) = MKB0
              CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)
*       JOINT ONE EQUATIONS---
*       AA = AG1 + (WD1 X RA/G1) + W1 X (W1 X RA/G1)
                VECTA1(1) = WDX(1)
                VECTA1(2) = WDY(1)
                VECTA1(3) = WDZ(1)
                RAG1(1) = JX1 - LCOGX(1)
                RAG1(2) = JY1 - LCOGY(1)
                RAG1(3) = JZ1 - LCOGZ(1)
              CALL CPROD(VECTA1,RAG1,MIA1,MJA1,MKA1)
                VECTA1(1) = W1(1)
                VECTA1(2) = W1(2)
                VECTA1(3) = W1(3)
              CALL CPROD (VECTA1,RAG1,MIB1,MJB1,MKB1)
                VECTB1(1) = MIB1
                VECTB1(2) = MJB1
                VECTB1(3) = MKB1
              CALL CPROD (VECTA1,VECTB1,MIC1,MJC1,MKC1)
*       AB = AG2 + (WD2 X RB/G2) + W2 X (W2 X RB/G2)
                VECTA1(1) = WDX(2)
                VECTA1(2) = WDY(2)
                VECTA1(3) = WDZ(2)
                RBG2(1) = JX1 - LCOGX(2)
                RBG2(2) = JY1 - LCOGY(2)
                RBG2(3) = JZ1 - LCOGZ(2)
              CALL CPROD (VECTA1,RBG2,MIA2,MJA2,MKA2)
                VECTA1(1) = W2(1)
                VECTA1(2) = W2(2)
                VECTA1(3) = W2(3)
              CALL CPROD (VECTA1,RBG2,MIB2,MJB2,MKB2)
                VECTB1(1) = MIB2
                VECTB1(2) = MJB2
                VECTB1(3) = MKB2
              CALL CPROD (VECTA1,VECTB1,MIC2,MJC2,MKC2)
*       JOINT TWO EQUATIONS
*       AA = AG2 + (WD2 X RA/G2) + W2 X (W2 X RA/G2)
                VECTA2(1) = WDX(2)
                VECTA2(2) = WDY(2)
                VECTA2(3) = WDZ(2)
                RAG2(1) = JX2 - LCOGX(2)
                RAG2(2) = JY2 - LCOGY(2)
                RAG2(3) = JZ2 - LCOGZ(2)
              CALL CPROD (VECTA2,RAG2,MIA3,MJA3,MKA3)
                VECTA2(1) = W2(1)
                VECTA2(2) = W2(2)
                VECTA2(3) = W2(3)
              CALL CPROD (VECTA2,RAG2,MIB3,MJB3,MKB3)
                VECTB2(1) = MIB3
                VECTB2(2) = MJB3
                VECTB2(3) = MKB3
              CALL CPROD(VECTA2,VECTB2,MIC3,MJC3,MKC3)
*       AB = AG3 + (WD3 X RB/G3) + W3 X (W3 X RB/G3)
```

```
                VECTA2(1) = WDX(3)
                VECTA2(2) = WDY(3)
                VECTA2(3) = WDZ(3)
                RBG3(1) = JX2 - LCOGX(3)
                RBG3(2) = JY2 - LCOGY(3)
                RBG3(3) = JZ2 - LCOGZ(3)
              CALL CPROD (VECTA2,RBG3,MIA4,MKA4,MKA4)
                VECTA2(1) = W3(1)
                VECTA2(2) = W3(2)
                VECTA2(3) = W3(3)
              CALL CPROD (VECTA2,RBG3,MIB4,MJB4,MKB4)
                VECTB2(1) = MIB4
                VECTB2(2) = MJB4
                VECTB2(3) = MKB4
              CALL CPROD (VECTA2,VECTB2,MIC4,MJC4,MKC4)

*       SUM OF MOMENTS EQUATIONS
         DO 100 I = 1,3
*      COMPUTE HX,H DOT X,HY,H DOT Y, HZ,H DOT Z

                RX(I,1) = -L(I,1) * DCOS(DRCRAX(I))
                RX(I,2) =  L(I,2) * DCOS(DRCRAX(I))
                RY(I,1) = -L(I,1) * DCOS(DRCRAY(I))
                RY(I,2) =  L(I,2) * DCOS(DRCRAY(I))
                RZ(I,1) = -L(I,1) * DCOS(DRCRAZ(I))
                RZ(I,2) =  L(I,2) * DCOS(DRCRAZ(I))

         IXX(I,1) = MASS(I,1) * ((RY(I,1) * RY(I,1)) + (RZ(I,1) * RZ(I,1)))
         IXX(I,2) = MASS(I,2) * ((RY(I,2) * RY(I,2)) + (RZ(I,2) * RZ(I,2)))
         IXXT(I)  = IXX(I,1) + IXX(I,2)
         IF (IXXT(I) .LE. .020) THEN
         IXXT(I)  = .020
         ELSE
         IXXT(I) = IXXT(I)
         END IF
         IXY(I,1) = MASS(I,1) * RX(I,1) * RY(I,1)
         IXY(I,2) = MASS(I,2) * RX(I,2) * RY(I,2)
         IXYT(I)  = IXY(I,1) + IXY(I,2)
         IXZ(I,1) = MASS(I,1) * RZ(I,1) * RX(I,1)
         IXZ(I,2) = MASS(I,2) * RZ(I,2) * RX(I,2)
         IXZT(I)  = IXZ(I,1) + IXZ(I,2)
         HDX(1) = WDX(1) * IXX(I,1)- WDZ(I) * IXZ(I,1) - WDY(I) * IXY(I,1)
         HDX(2) = WDX(2) * IXX(I,2)- WDZ(I) * IXZ(I,2) - WDY(I) * IXY(I,2)
         IYY(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RZ(I,1) * RZ(I,1)))
         IYY(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RZ(I,2) * RZ(I,2)))
         IYYT(I)  = IYY(I,1) + IYY(I,2)
         IF (IYYT(I) .LE. .020) THEN
         IYYT(I)  = .020
         ELSE
         IYYT(I)  = IYYT(I)
         END IF
         IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
         IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
         IYZT(I)  = IYZ(I,1) + IYZ(I,2)
         HDY(1) = WDY(I) * IYY(I,1) - WDX(I) * IXY(I,1) - WDZ(I) * IYZ(I,1)
         HDY(2) = WDY(I) * IYY(I,2) - WDX(I) * IXY(I,2) - WDZ(I) * IYZ(I,2)
         IZZ(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RY(I,1) * RY(I,1)))
         IZZ(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RY(I,2) * RY(I,2)))
         IZZT(I)  = IZZ(I,1) + IZZ(I,2)
         IF (IZZT(I) .LE. .020) THEN
```

```
      IZZT(I)  = .020
      ELSE
      IZZT(I)  = IZZT(I)
      END IF
      HDZ(1) = WDZ(I) * IZZ(I,1) - WDX(I) * IXZ(I,1) - WDY(I) * IYZ(I,1)
      HDZ(2) = WDZ(I) * IZZ(I,2) - WDX(I) * IXZ(I,2) - WDY(I) * IYZ(I,2)

                SUMHDX(I) = HDX(1) + HDX(2)
                SUMHDY(I) = HDY(1) + HDY(2)
                SUMHDZ(I) = HDZ(1) + HDZ(2)
 100    CONTINUE
*      ENTER CONSTANTS INTO MATRIX A
*      LINK ONE
*      SUM OF FORCES IN THE X DIRECTION
           MATA(1,1)  =  1.0D0
           MATA(1,4)  =  MASS1
           MATA(1,10) = -1.0D0
           MATB(1)    =  0.0D0
*      SUM OF FORCES IN Y DIRECTION
           MATA(2,2)  =  1.0D0
           MATA(2,5)  =  MASS1
           MATA(2,11) = -1.0D0
           MATB(2)    =  0.0D0
*      SUM OF FORCES IN Z DIRECTION
           MATA(3,3)  =  1.0D0
           MATA(3,6)  =  MASS1
           MATA(3,12) = -1.0D0
*      SUM OF FORCES LINK ONE EQUAL
           MATB(3)    = -WG1
*      EQUATIONS AT JOINT ZERO
*      IN THE X DIRECTION
           MATA(4,4) =  1.0D0
           MATA(4,8) =  RBG1(3)
           MATA(4,9) = -RBG1(2)
           MATB(4)   =  A0X - MIC0
*      IN THE Y DIRECTION
           MATA(5,5) =  1.0D0
           MATA(5,7) = -RBG1(3)
           MATA(5,9) =  RBG1(1)
           MATB(5)   =  A0Y - MJC0
*      IN THE Z DIRECTION
           MATA(6,6) =  1.0D0
           MATA(6,7) =  RBG1(2)
           MATA(6,8) = -RBG1(1)
           MATB(6)   =  A0Z - MKC0

*      SUM OF MOMENTS EQUATIONS FOR LINK ONE IN THE X,Y,Z DIRECTIONS
           MATA(7,2)  =  RBG1(3)
           MATA(7,3)  = -RBG1(2)
           MATA(7,7)  = -IXXT(1)
           MATA(7,8)  =  IXYT(1)
           MATA(7,9)  =  IXZT(1)
           MATA(7,11) = -RAG1(3)
           MATA(7,12) =  RAG1(2)
           MATB(7)    =  T1X - T0X
           MATA(8,1)  = -RBG1(3)
```

```
      MATA(8,3)  =  RBG1(1)
      MATA(8,7)  =  IXYT(1)
      MATA(8,8)  = -IYYT(1)
      MATA(8,9)  =  IYZT(1)
      MATA(8,10) =  RAG1(3)
      MATA(8,12) = -RAG1(1)
      MATB(8)    =  T1Y - T0Y
      MATA(9,1)  =  RBG1(2)
      MATA(9,2)  = -RBG1(1)
      MATA(9,7)  =  IXZT(1) + IXZT(2) + IXZT(3)
      MATA(9,8)  =  IYZT(1) + IYZT(2) + IYZT(3)
      MATA(9,9)  = -IZZT(1) - IZZT(2) - IZZT(3)
      MATA(9,10) = -RAG1(2)
      MATA(9,11) =  RAG1(1)
      MATB(9)    =  T1Z - T0Z
*     LINK TWO
*     SUM OF FORCES IN X DIRECTION
      MATA(10,10) =  1.0D0
      MATA(10,13) =  MASS2
      MATA(10,19) = -1.0D0
      MATB(10)    =  0.0D0
*     SUM OF FORCES IN THE Y DIRECTION
      MATA(11,11) =  1.0D0
      MATA(11,14) =  MASS2
      MATA(11,20) = -1.0D0
      MATB(11)    =  0.0D0
*     SUM OF FORCES IN THE Z DIRECTION
      MATA(12,12) =  1.0D0
      MATA(12,15) =  MASS2
      MATA(12,21) = -1.0D0
*     SUM OF FORCES LINK TWO EQUAL
      MATB(12)    = -WG2
*     EQUATIONS AT JOINT ONE
*     IN THE X DIRECTION
      MATA(13,4)  = -1.0D0
      MATA(13,8)  = -RAG1(3)
      MATA(13,9)  =  RAG1(2)
      MATA(13,13) =  1.0D0
      MATA(13,17) =  RBG2(3)
      MATA(13,18) = -RBG2(2)
      MATB(13)    =  MIC1 - MIC2
*     IN THE Y DIRECTION
      MATA(14,5)  = -1.0D0
      MATA(14,7)  =  RAG1(3)
      MATA(14,9)  = -RAG1(1)
      MATA(14,14) =  1.0D0
      MATA(14,16) = -RBG2(3)
      MATA(14,18) =  RBG2(1)
      MATB(14)    =  MJC1 - MJC2
*     IN THE Z DIRECTION
      MATA(15,6)  = -1.0D0
      MATA(15,7)  = -RAG1(2)
      MATA(15,8)  =  RAG1(1)
      MATA(15,15) =  1.0D0
      MATA(15,16) =  RBG2(2)
      MATA(15,17) = -RBG2(1)
      MATB(15)    =  MKC1 - MKC2
*     SUM OF MOMENTS EQUATIONS FOR LINK TWO IN THE X,Y,Z DIRECTIONS
```

```
            MATA(16,11) =  RBG2(3)
            MATA(16,12) = -RBG2(2)
            MATA(16,16) = -IXXT(2)
            MATA(16,17) =  IXYT(2)
            MATA(16,18) =  IXZT(2)
            MATA(16,20) = -RAG2(3)
            MATA(16,21) =  RAG2(2)
            MATB(16)    =  (-T1X + T2X) * DCOS(RLRZ1)
            MATA(17,10) = -RBG2(3)
            MATA(17,12) =  RBG2(1)
            MATA(17,16) =  IXYT(2)
            MATA(17,17) = -IYYT(2)
            MATA(17,18) =  IYZT(2)
            MATA(17,19) =  RAG2(3)
            MATA(17,21) = -RAG2(1)
            MATB(17)    =  (- T1Y + T2Y) * DSIN(RLRZ1)
            MATA(18,9)  = -1.0D0
            MATA(18,18) =  1.0D0
            MATB(18)    =  0.0D0
*         MATA(18,10) =  RBG2(2)
*         MATA(18,11) = -RBG2(1)
*         MATA(18,16) =  IXZT(2) + IXZT(3)
*         MATA(18,17) =  IYZT(2) + IYZT(3)
*         MATA(18,18) = -IZZT(2) - IZZT(3)
*         MATA(18,19) = -RAG2(2)
*         MATA(18,20) =  RAG2(1)
*         MATB(18) =  - T1Z + T2Z
*       LINK THREE
*       SUM OF FORCES IN THE X DIRECTION
            MATA(19,19) =  1.0D0
            MATA(19,22) =  MASS3
            MATB(19)    =  0.0D0
*       SUM OF FORCES IN THE Y DIRECTION
            MATA(20,20) =  1.0D0
            MATA(20,23) =  MASS3
            MATB(20)    =  0.0D0
*       SUM OF FORCES IN THE Z DIRECTION
            MATA(21,21) =  1.0D0
            MATA(21,24) =  MASS3
            MATB(21) = -WG3
*       EQUATIONS AT JOINT TWO
*       IN THE X DIRECTION
            MATA(22,13) = -1.0D0
            MATA(22,17) = -RAG2(3)
            MATA(22,18) =  RAG2(2)
            MATA(22,22) =  1.0D0
            MATA(22,26) =  RBG3(3)
            MATA(22,27) = -RBG3(2)
            MATB(22)    =  MIC3 - MIC4
*       IN THE Y DIRECTION
            MATA(23,14) = -1.0D0
            MATA(23,16) =  RAG2(3)
            MATA(23,18) = -RAG2(1)
            MATA(23,23) =  1.0D0
            MATA(23,25) = -RBG3(3)
            MATA(23,27) =  RBG3(1)
            MATB(23)    =  MJC3 - MJC4
```

```
*       IN THE Z DIRECTION
            MATA(24,15) = -1.0D0
            MATA(24,16) = -RAG2(2)
            MATA(24,17) =  RAG2(1)
            MATA(24,24) =  1.0D0
            MATA(24,25) =  RBG3(2)
            MATA(24,26) = -RBG3(1)
            MATB(24)    =  MKC3 - MKC4

*       SUM OF MOMENTS EQUATIONS FOR LINK THREE IN THE X,Y,Z DIRECTIONS
            MATA(25,20) =  RBG3(3)
            MATA(25,21) = -RBG3(2)
            MATA(25,25) = -IXXT(3)
            MATA(25,26) =  IXYT(3)
            MATA(25,27) =  IXZT(3)
            MATB(25)    = -T2X * DCOS(RLRZ1)
            MATA(26,19) = -RBG3(3)
            MATA(26,21) =  RBG3(1)
            MATA(26,25) =  IXYT(3)
            MATA(26,26) = -IYYT(3)
            MATA(26,27) =  IYZT(3)
            MATB(26)    = -T2Y * DSIN(RLRZ1)
            MATA(27,9)  = -1.0D0
            MATA(27,27) =  1.0D0
            MATB(27)    =  0.0D0
*         MATA(27,19) =  RBG3(2)
*         MATA(27,20) = -RBG3(1)
*         MATA(27,25) =  IXZT(3)
*         MATA(27,26) =  IYZT(3)
*         MATA(27,27) = -IZZT(3)
*
*         MATB(27)    = - T2Z
            GO TO 1112
*       INITIALIZE MATRIX ACCORDING TO CONSTRAINTS
*       CONSTRAINTS GROUP 1 WHEN ONLY LINK THREE IS IN MOTION
*         DO 118 I = 1,18
*           DO 18 J = 1,27
*             MATA(I,J) = 0.0
*             MATA(I,I) = 1.0
*             MATB(I)   = 0.0
*18         CONTINUE
*118      CONTINUE
*
*         DO 181 I = 19,27
*           DO 81 J = 1,18
*             MATA(I,J) = 0.0
*81         CONTINUE
*181         CONTINUE
*       GO TO 1111
*       CONSTRAINTS GROUP 2 WHEN LINK TWO AND THREE ARE IN MOTION
*         DO 19 I = 1,9
*           DO 191 J = 1,27
*             MATA(I,J) = 0.0D0
*             MATA(I,I) = 1.0 D0
*             MATB(I)   = 0.0D0
*
*             mATA(17,J) = 0.0D0
*             MATA(18,J) = 0.0D0
*             MATB(17)   = 0.0D0
*             MATB(18)   = 0.D0
*
*             MATA(J,17) = 0.0D0
```

```
*            MATA(J,18) = 0.0D0
*
*            mATA(17,17) = 1.0D0
*            MATA(18,18) = 1.0D0
*
*191       CONTINUE
*19      CONTINUE
*
*        DO 91 I = 10,27
*          DO 92 J = 1,9
*            MATA(I,J) = 0.0
*92        CONTINUE
*91      CONTINUE
**       GO TO 1111
*      CONSTRAINTS GROUP 3 WHEN THREE OF THE LINKS  ARE IN MOTION
*        DO 78 J = 1,27
*          MATA(7,J)  = 0.0D0
*          MATA(8,J)  = 0.0D0
*          MATA(J,7)  = 0.0D0
*          MATA(J,8)  = 0.0D0
*          MATB(7)    = 0.0D0
*          MATB(8)    = 0.0D0
*          MATA(17,J) = 0.0D0
*          MATA(18,J) = 0.0D0
*          MATA(J,17) = 0.0D0
*          MATA(J,18) = 0.0D0
*          MATB(17)   = 0.0D0
*          MATB(18)   = 0.0D0
*          MATA(26,J) = 0.0D0
*          MATA(27,J) = 0.0D0
*          MATA(J,26) = 0.0D0
*          MATA(J,27) = 0.0D0
*          MATB(26)   = 0.0D0
*          MATB(27)   = 0.0D0
*          MATA(7,7)   = 1.0D0
*          MATA(8,8)   = 1.0D0
*          MATA(17,17) = 1.0D0
*          MATA(18,18) = 1.0D0
*          MATA(26,26) = 1.0D0
*          MATA(27,27) = 1.0D0
*78       CONTINUE

*  CONSTRAINT GROUP 4  THE FIRST LINK IS CONSTRAINED TO ROTATE IN Z DIR.
 1112    DO 48 I = 4,8
             DO 481 J = 1,27
               MATA(I,J) = 0.0D0
               MATA(I,I) = 1.0 D0
               MATB(I)   = 0.0D0
 481     CONTINUE
 48      CONTINUE
           DO 84 I = 9,27
             DO 841 J = 4,8
               MATA(I,J) = 0.0
 841        CONTINUE
 84      CONTINUE
*      CALL EQUATION SOLVER PROGRAM FROM IMSL
 1111  CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
*      IF (IER .NE. 0) CALL ENDJOB

*      FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
*      LINK ONE
           AX1       = MATB(4)
```

```
      VELX1     = INTGRL(0.,AX1)
      LCOGX1    = INTGRL(X1,VELX1)
      LCOGX(1)  = LCOGX1
      AY1       = MATB(5)
      VELY1     = INTGRL(0.,AY1)
      LCOGY1    = INTGRL(Y1,VELY1)
      LCOGY(1)  = LCOGY1
      AZ1       = MATB(6)
      VELZ1     = INTGRL(0.,AZ1)
      LCOGZ1    = INTGRL(Z1,VELZ1)
      LCOGZ(1)  = LCOGZ1
      WD1X      = MATB(7)
      W1X       = INTGRL(0.,WD1X)
      WDX(1)    = WD1X
      W1(1)     = W1X
      WD1Y      = MATB(8)
      W1Y       = INTGRL(0.,WD1Y)
      WDY(1)    = WD1Y
      W1(2)     = W1Y
      WD1Z      = MATB(9)
      W1Z       = INTGRL(0.,WD1Z)
      WDZ(1)    = WD1Z
      W1(3)     = W1Z
*     TRANSFORMATION MATRIX FROM EARTH FIXED TO BODY FIXED COORDINATE
*     SYSTEM FOR LINK ONE
      MAT1R(1,1) = DCOS(RLRZ1) * DCOS(PTRY1)
      MAT1R(2,1) = DCOS(RLRZ1) * DSIN(PTRY1) * DSIN(YWRX1) -...
      DSIN(RLRZ1) * DCOS(YWRX1)
      MAT1R(3,1) = DCOS(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) +...
      DSIN(RLRZ1) * DSIN(YWRX1)
      MAT1R(1,2) = DSIN(RLRZ1)*DCOS(PTRY1)
      MAT1R(2,2) = DSIN(RLRZ1) * DSIN(PTRY1) * DSIN(YWRX1) +...
      DCOS(RLRZ1) * DCOS(YWRX1)
      MAT1R(3,2) = DSIN(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) -...
      DCOS(RLRZ1) * DSIN(YWRX1)
      MAT1R(1,3) = -DSIN(PTRY1)
      MAT1R(2,3) = DCOS(PTRY1) * DSIN(YWRX1)
      MAT1R(3,3) = DCOS(PTRY1) * DCOS(YWRX1)
*     GET THE VELOCITIES OF LINK ONE IN BODY FIXED COORDINATE SYSTEM
      DO 605 J = 1,3
        SUM1 = 0.0D0
          DO 606 K = 1,3
            SUM1 = SUM1 + MAT1R(J,K) * W1(K)
606       CONTINUE
          BRATE1(J) = SUM1
605   CONTINUE
*     TRANSFORMATION MATRIX FROM BODY FIXED TO NON-ORTHOGONAL EULER
*     COORDINATE SYSTEM FOR LINK ONE
      MAT1T(1,1) = 0.0D0
      MAT1T(2,1) = 1.0D0
      MAT1T(3,1) = 0.0D0
      MAT1T(1,2) =  DCOS(YWRX1)
      MAT1T(2,2) =  DTAN(PTRY1) * DSIN(YWRX1)
      MAT1T(3,2) =  1.0D0/DCOS(PTRY1) * DSIN(YWRX1)
      MAT1T(1,3) = -DSIN(YWRX1)
      MAT1T(2,3) =  DTAN(PTRY1) * DCOS(YWRX1)
      MAT1T(3,3) =  1.D0/DCOS(PTRY1) * DCOS(YWRX1)
*     GET THE VELOCITIES OF LINK ONE IN THE EULER COORDINATE SYSTEM
```

```
            DO 705 J = 1,3
               SUM1 = 0.0D0
                  DO 706 K = 1,3
                     SUM1 = SUM1 + MAT1T(J,K) * BRATE1(K)
 706              CONTINUE
                  RATE1(J) = SUM1
 705        CONTINUE
            RATE1X = RATE1(1)
            RATE1Y = RATE1(2)
            RATE1Z = RATE1(3)
*       INTEGRATION OF THE VELOCITIES OF LINK ONE IN EULER COOR. SYSTEM
            YWRX1 = INTGRL(0.,RATE1X)
            PTRY1 = INTGRL(0.,RATE1Y)
            RLRZ1 = INTGRL(-PI/2.,RATE1Z)
*       CONVERT THE ANGLES TO DEGREES
            YAWANX(1) = YWRX1 * RADEG
            PTCANY(1) = PTRY1 * RADEG
            ROLANZ(1) = RLRZ1 * RADEG
*       GET THE DIRECTION COSINES FOR THE LINK ONE
            DRCSY(1) = DCOS(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) +...
            DSIN(RLRZ1) * DSIN(YWRX1)
            DRCSX(1) = DSIN(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) -...
            DCOS(RLRZ1) * DSIN(YWRX1)
            DRCSZ(1) = DCOS(PTRY1) * DCOS(YWRX1)
            DRCRAX(1) = DACOS(DRCSX(1))
            DRCRAY(1) = DACOS(DRCSY(1))
            DRCRAZ(1) = DACOS(DRCSZ(1))
            DRCANX(1) = DACOS(DRCSX(1)) * RADEG
            DRCANY(1) = DACOS(DRCSY(1)) * RADEG
            DRCANZ(1) = DACOS(DRCSZ(1)) * RADEG

*       LINK TWO
9             AX2       = MATB(13)
              VELX2     = INTGRL(0.,AX2)
              LCOGX2    = INTGRL(X2,VELX2)
              LCOGX(2)  = LCOGX2
              AY2       = MATB(14)
              VELY2     = INTGRL(0.,AY2)
              LCOGY2    = INTGRL(Y2,VELY2)
              LCOGY(2)  = LCOGY2
              AZ2       = MATB(15)
              VELZ2     = INTGRL(0.,AZ2)
              LCOGZ2    = INTGRL(Z2,VELZ2)
              LCOGZ(2)  = LCOGZ2
              WD2X      = MATB(16)
              W2X       = INTGRL(0.,WD2X)
              WDX(2)    = WD2X
              W2(1)     = W2X
              WD2Y      = MATB(17)
              W2Y       = INTGRL(0.,WD2Y)
              WDY(2)    = WD2Y
              W2(2)     = W2Y
              WD2Z      = MATB(18)
              W2Z       = INTGRL(0.,WD2Z)
              WDZ(2)    = WD2Z
              W2(3)     = W2Z
*       TRANSFORMATION MATRIX FROM EARTH FIXED TO BODY FIXED COORDINATE
*       SYSTEM FOR LINK TWO
            MAT2R(1,1) = DCOS(RLRZ2) * DCOS(PTRY2)
```

```
      MAT2R(2,1) = DCOS(RLRZ2) * DSIN(PTRY2) *  DSIN(YWRX2) -...
      DSIN(RLRZ2) * DCOS(YWRX2)
      MAT2R(3,1) = DCOS(RLRZ2) * DSIN(PTRY2) * DCOS(YWRX2) +...
      DSIN(RLRZ2) * DSIN(YWRX2)
      MAT2R(1,2) = DSIN(RLRZ2) * DCOS(PTRY2)
      MAT2R(2,2) = DSIN(RLRZ2) * DSIN(PTRY2) * DSIN(YWRX2) +...
      DCOS(RLRZ2) * DCOS(YWRX2)
      MAT2R(3,2) = DSIN(RLRZ2) * DSIN(PTRY2) * DCOS(YWRX2) -...
      DCOS(RLRZ2) * DSIN(YWRX2)
      MAT2R(1,3) = -DSIN(PTRY2)
      MAT2R(2,3) = DCOS(PTRY2) * DSIN(YWRX2)
      MAT2R(3,3) = DCOS(PTRY2) * DCOS(YWRX2)
*     GET THE VELOCITIES OF LINK TWO IN BODY FIXED COORDINATE SYSTEM
      DO 607 J = 1,3
        SUM1 = 0.0D0
          DO 608 K = 1,3
            SUM1 = SUM1 + MAT2R(J,K) * W2(K)
608     CONTINUE
        BRATE2(J) = SUM1
607   CONTINUE
*     TRANSFORMATION MATRIX FROM BODY FIXED TO NON-ORTHOGONAL EULER
*     COORDINATE SYSTEM FOR LINK TWO
      MAT2T(1,1) = 0.0D0
      MAT2T(2,1) = 1.0D0
      MAT2T(3,1) = 0.0D0
      MAT2T(1,2) =  DCOS(YWRX2)
      MAT2T(2,2) =  DTAN(PTRY2) * DSIN(YWRX2)
      MAT2T(3,2) =  1.0D0/DCOS(PTRY2) * DSIN(YWRX2)
      MAT2T(1,3) = -DSIN(YWRX2)
      MAT2T(2,3) =  DTAN(PTRY2) * DCOS(YWRX2)
      MAT2T(3,3) =  1.0D0/DCOS(PTRY2) * DCOS(YWRX2)
*     GET THE VELOCITIES OF LINK TWO IN THE EULER COORDINATE SYSTEM
      DO 707 J = 1,3
        SUM1 = 0.0D0
          DO 708 K = 1,3
            SUM1 = SUM1 + MAT2T(J,K) * BRATE2(K)
708     CONTINUE
        RATE2(J) = SUM1
707   CONTINUE
      RATE2X = RATE2(1)
      RATE2Y = RATE2(2)
      RATE2Z = RATE2(3)
*     INTEGRATION OF THE VELOCITIES OF LINK TWO IN EULER COOR. SYSTEM
      YWRX2 = INTGRL(0.,RATE2X)
      PTRY2 = INTGRL(0.,RATE2Y)
      RLRZ2 = INTGRL(-PI/2.,RATE2Z)
*     CONVERT THE ANGLES TO DEGREES
      YAWANX(2) = YWRX2 * RADEG
      PTCANY(2) = PTRY2 * RADEG
      ROLANZ(2) = RLRZ2 * RADEG
*     GET THE DIRECTION COSINES FOR THE LINK TWO
      DRCSY(2) = DCOS(RLRZ2) * DSIN(PTRY2) * DSIN(YWRX2) -...
      DSIN(RLRZ2) * DCOS(YWRX2)
      DRCSX(2) = DSIN(RLRZ2) * DSIN(PTRY2)*DSIN(YWRX2) +...
      DCOS(RLRZ2) * DCOS(YWRX2)
      DRCSZ(2) = DCOS(PTRY2) * DSIN(YWRX2)
      DRCRAX(2) = DACOS(DRCSX(2))
```

```
      DRCRAY(2) = DACOS(DRCSY(2))
      DRCRAZ(2) = DACOS(DRCSZ(2))
      DRCANX(2) = DACOS(DRCSX(2)) * RADEG
      DRCANY(2) = DACOS(DRCSY(2)) * RADEG
      DRCANZ(2) = DACOS(DRCSZ(2)) * RADEG
      JX1  =  (L(1,1) + L(1,2)) * DCOS(DRCRAX(1))
      JY1  =  (L(1,1) + L(1,2)) * DCOS(DRCRAY(1))
      JZ1  =  (L(1,1) + L(1,2)) * DCOS(DRCRAZ(1))

*     LINK THREE
6       AX3       = MATB(22)
        VELX3     = INTGRL(0.,AX3)
        LCOGX3    = INTGRL(X3,VELX3)
        LCOGX(3)  = LCOGX3
        AY3       = MATB(23)
        VELY3     = INTGRL(0.,AY3)
        LCOGY3    = INTGRL(Y3,VELY3)
        LCOGY(3)  = LCOGY3
        AZ3       = MATB(24)
        VELZ3     = INTGRL(0.,AZ3)
        LCOGZ3    = INTGRL(Z3,VELZ3)
        LCOGZ(3)  = LCOGZ3
        WD3X      = MATB(25)
        W3X       = INTGRL(0.,WD3X)
        WDX(3)    = WD3X
        W3(1)     = W3X
        WD3Y      = MATB(26)
        W3Y       = INTGRL(0.,WD3Y)
        WDY(3)    = WD3Y
        W3(2)     = W3Y
        WD3Z      = MATB(27)
        W3Z       = INTGRL(0.,WD3Z)
        WDZ(3)    = WD3Z
        W3(3)     = W3Z
*     TRANSFORMATION MATRIX FROM EARTH FIXED TO BODY FIXED COORDINATE
*     SYSTEM FOR LINK THREE
      MAT3R(1,1) = DCOS(RLRZ3) * DCOS(PTRY3)
      MAT3R(2,1) = DCOS(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) -...
      DSIN(RLRZ3) * DCOS(YWRX3)
      MAT3R(3,1) = DCOS(RLRZ3) * DSIN(PTRY3) * DCOS(YWRX3) +...
      DSIN(RLRZ3) * DSIN(YWRX3)
      MAT3R(1,2) = DSIN(RLRZ3) * DCOS(PTRY3)
      MAT3R(2,2) = DSIN(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) +...
      DCOS(RLRZ3) * DCOS(YWRX3)
      MAT3R(3,2) = DSIN(RLRZ3) * DSIN(PTRY3) * DCOS(YWRX3) -...
      DCOS(RLRZ3) * DSIN(YWRX3)
      MAT3R(1,3) = -DSIN(PTRY3)
      MAT3R(2,3) = DCOS(PTRY3) * DSIN(YWRX3)
      MAT3R(3,3) = DCOS(PTRY3) *DCOS(YWRX3)
*     GET THE VELOCITIES OF LINK THREE IN BODY FIXED COORDINATE SYSTEM
      DO 609 J = 1,3
        SUM1 = 0.0D0
          DO 610 K = 1,3
            SUM1 = SUM1 + MAT3R(J,K) * W3(K)
610       CONTINUE
          BRATE3(J) = SUM1
609   CONTINUE
*     TRANSFORMATION MATRIX FROM BODY FIXED TO NON-ORTHOGONAL EULER
*     COORDINATE SYSTEM FOR LINK THREE
```

```
            MAT3T(1,1) = 0.0D0
            MAT3T(2,1) = 1.0D0
            MAT3T(3,1) = 0.0D0

            MAT3T(1,2) =  DCOS(YWRX3)
            MAT3T(2,2) =  DTAN(PTRY3) * DSIN(YWRX3)
            MAT3T(3,2) =  1.0D0/DCOS(PTRY3) * DSIN(YWRX3)

            MAT3T(1,3) = -DSIN(YWRX3)
            MAT3T(2,3) =  DTAN(PTRY3) * DCOS(YWRX3)
            MAT3T(3,3) =  1.0D0/DCOS(PTRY3) * DCOS(YWRX3)
*       GET THE VELOCITIES OF LINK THREE IN THE EULER COORDINATE SYSTEM
            DO 709 J = 1,3
               SUM1 = 0.0D0
                  DO 710 K = 1,3
                     SUM1 = SUM1 + MAT3T(J,K) * BRATE3(K)
 710           CONTINUE
               RATE3(J) = SUM1
 709        CONTINUE

            RATE3X = RATE3(1)
            RATE3Y = RATE3(2)
            RATE3Z = RATE3(3)
*       INTEGRATION OF THE VELOCITIES OF LINK THREE IN EULER COOR. SYSTEM
            YWRX3 = INTGRL(0.,RATE3X)
            PTRY3 = INTGRL(0.,RATE3Y)
            RLRZ3 = INTGRL(-PI/2.,RATE3Z)

*       CONVERT THE ANGLES TO DEGREES
            YAWANX(3) = YWRX3 * RADEG
            PTCANY(3) = PTRY3 * RADEG
            ROLANZ(3) = RLRZ3 * RADEG
*       GET THE DIRECTION COSINES FOR THE LINK THREE
            DRCSY(3) = DCOS(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) -...
            DSIN(RLRZ3) * DCOS(YWRX3)
            DRCSX(3) = DSIN(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) +...
            DCOS(RLRZ3) * DCOS(YWRX3)
            DRCSZ(3) = DCOS(PTRY3) * DSIN(YWRX3)

            DRCRAX(3) = DACOS(DRCSX(3))
            DRCRAY(3) = DACOS(DRCSY(3))
            DRCRAZ(3) = DACOS(DRCSZ(3))
            DRCANX(3) = DACOS(DRCSX(3)) * RADEG
            DRCANY(3) = DACOS(DRCSY(3)) * RADEG
            DRCANZ(3) = DACOS(DRCSZ(3)) * RADEG
            JX2 = JX1 + (L(2,1) + L(2,2)) * DCOS(DRCRAX(2))
            JY2 = JY1 + (L(2,1) + L(2,2)) * DCOS(DRCRAY(2))
            JZ2 = JZ1 + (L(2,1) + L(2,2)) * DCOS(DRCRAZ(2))
            TIPX = JX2 + (L(3,1) + L(3,2)) * DCOS(DRCRAX(3))
            TIPY = JY2 + (L(3,1) + L(3,2)) * DCOS(DRCRAY(3))
            TIPZ = JZ2 + (L(3,1) + L(3,2)) * DCOS(DRCRAZ(3))

DYNAMIC
NOSORT
*       INPUT TORQUE AT JOINTS
                  T0Z =  A * SIN (W * TIME + P)
                  T1X =-10 * SIN (W * TIME + P)
                  T2X =  A * SIN (W * TIME + P)

END
STOP
FORTRAN
```

```
*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS
      SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
          IMPLICIT REAL*8 (A-Z)
          DIMENSION VECTA(3),VECTB(3)
          MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
          MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
          MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
       RETURN
       END
```

APPENDIX C

THREE DIMENSIONAL SIMULATION PROGRAM
INVESTIGATION OF SINGULAR CONFIGURATION

```
TERMINAL
METHOD ADAMS
PRINT .05,ERROR,ANG12Z,ANG23Z
CONTROL FINTIM = 4.0, DELMAX =.1, DELPRT = .05
SAVE .05, ERROR,ANG12X,ANG12Y,ANG12Z,THETAB,THETAD,ANG23X,ANG23Y,...
ANG23Z,IYYT(2),IXXT(2),IZZT(2)
GRAPH(DE=TEK618) TIME,ANG12X
GRAPH(DE=TEK618) TIME,ANG12Y
GRAPH(DE=TEK618) TIME,ANG12Z
GRAPH(DE=TEK618) TIME,ANG23X
GRAPH(DE=TEK618) TIME,ANG23Y
GRAPH(DE=TEK618) TIME,ANG23Z
GRAPH(DE=TEK618) TIME,THETAB
GRAPH(DE=TEK618) TIME,IYYT(2),IXXT(2),IZZT(2)
D     DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D     DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D     DIMENSION MAT1R(3,3),MAT2R(3,3),MAT3R(3,3)
D     DIMENSION MAT1T(3,3),MAT2T(3,3),MAT3T(3,3)
D     INTEGER IER,I,J,M,K,P,N,IA,IDGT,A
EXCLUDE IA,IDGT,IER,I,J,M,K,P,N,A
ARRAY MATB(27),LCOGX(3),LCOGY(3),LCOGZ(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY WDX(3),WDY(3),WDZ(3),W1(3),W2(3),W3(3),MATC(27),DQ(27)
ARRAY RATE1(3),RATE2(3),RATE3(3),BRATE1(3),BRATE2(3),BRATE3(3)
ARRAY RBG1(3),RAG1(3),RBG2(3),RAG2(3),RBG3(3)
ARRAY SUMHDX(3),SUMHDY(3),SUMHDZ(3),HDX(2),HDY(2),HDZ(2),WKAREA(850)
ARRAY IXXT(3),IYYT(3),IZZT(3),IXYT(3),IXZT(3),IYZT(3)
ARRAY YAWANX(3),PTCANY(3),ROLANZ(3)
ARRAY DRCANX(3),DRCANY(3),DRCANZ(3)
ARRAY DRCRAX(3),DRCRAY(3),DRCRAZ(3)
ARRAY DRCSX(3),DRCSY(3),DRCSZ(3)
D     DATA MATA/729 * 0.0D0/

INITIAL
*       INPUT PARAMETER CONSTANTS
            A = 3.0D0
            P = 0.0D0
            W = PI / 2.0D0
            IDGT = 3
            G=0.0D0
            N=27
            M=1
            IA =27
*       INPUT JOINT LOCATIONS IN METERS
            JX0 = 0.0D0
            JY0 = 0.0D0
            JZ0 = 0.0D0
            JX1 = 0.0D0
            JY1 = 0.0D0
            JZ1 = 1.0D0
*       USE THE NEXT SET OF JOINT TWO COORDINATES FOR CASE A
            JX2 = 0.0D0
            JY2 = 1.0D0
            JZ2 = 1.0D0
*       USE THE NEXT SET OF JOINT TWO COORDINATES FOR CASE B
```

```
*          JX2 = 1.0D0
*          JY2 = 0.0D0
*          JZ2 = 1.0D0
*       USE THE NEXT SET OF JOINT TWO COORDINATES FOR CASE C
*
*          JX2 = 0.0D0
*          JY2 = 0.0D0
*          JZ2 = 2.0D0
*       INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS FOR
*       EACH LINK ENDS
           L(1,1) = 0.50D0
           L(1,2) = 0.50D0
           L(2,1) = 0.50D0
           L(2,2) = 0.50D0
           L(3,1) = 0.50D0
           L(3,2) = 0.50D0
*       INPUT MASS AT LINK ENDS IN KILOGRAMS
           MASS(1,1) = 2.5D0
           MASS(1,2) = 2.5D0
           MASS(2,1) = 2.5D0
           MASS(2,2) = 2.5D0
           MASS(3,1) = 2.5D0
           MASS(3,2) = 2.5D0
*       INPUT OMEGA AND OMEGA DOT
           DO 30 I = 1,3
                W1(I)     = 0.0D0
                W2(I)     = 0.0D0
                W3(I)     = 0.0D0
                WDX(I)    = 0.0D0
                WDY(I)    = 0.0D0
                WDZ(I)    = 0.0D0
30         CONTINUE
*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
*       LINK ONE
           LCOGX(1) = 0.0D0
           X1   =  LCOGX(1)
           LCOGY(1) = 0.0D0
           Y1   =  LCOGY(1)
           LCOGZ(1) = 0.5D0
           Z1   =  LCOGZ(1)
*       NEXT SET FOR LINK TWO AND THREE TO USE  FOR CASE A
            LCOGX(2) = 0.0D0
            X2   =  LCOGX(2)
            LCOGY(2) = 0.5D0
            Y2   =  LCOGY(2)
            LCOGZ(2) = 1.0D0
            Z2   =  LCOGZ(2)
            LCOGX(3) = 0.0D0
            X3   =  LCOGX(3)
            LCOGY(3) = 1.5D0
            Y3   =  LCOGY(3)
            LCOGZ(3) = 1.0D0
            Z3   =  LCOGZ(3)
*       NEXT SET FOR LINK TWO AND THREE TO USE  FOR CASE B
*           LCOGX(2) = 0.5D0
*           X2   =  LCOGX(2)
*           LCOGY(2) = 0.0D0
*           Y2   =  LCOGY(2)
*           LCOGZ(2) = 1.0D0
*           Z2   =  LCOGZ(2)
*           LCOGX(3) = 1.5D0
*           X3   =  LCOGX(3)
```

```
*          LCOGY(3) = 0.0D0
*          Y3   =  LCOGY(3)
*          LCOGZ(3) = 1.0D0
*          Z3   =  LCOGZ(3)
*      NEXT SET FOR LINK TWO AND THREE TO USE  FOR CASE C
*          LCOGX(2) = 0.0D0
*          X2   =  LCOGX(2)
*          LCOGY(2) = 0.0D0
*          Y2   =  LCOGY(2)
*          LCOGZ(2) = 1.5D0
*          Z2   =  LCOGZ(2)
*          LCOGX(3) = 0.0D0
*          X3   =  LCOGX(3)
*          LCOGY(3) = 0.0D0
*          Y3   =  LCOGY(3)
*          LCOGZ(3) = 2.5D0
*          Z3   =  LCOGZ(3)
*      INPUT MASS OF EACH LINK IN KG AND COMPUTE WEIGHTS IN NEWTONS
           MASS1 = 5.0D0
           MASS2 = 5.0D0
           MASS3 = 5.0D0
           WG1 = MASS1*G
           WG2 = MASS2*G
           WG3 = MASS3*G
*      INPUT ACCELERATION OF JOINT ZERO
           A0X = 0.0D0
           A0Y = 0.0D0
           A0Z = 0.0D0
*      INITIALIZE MATRIX A AND B TO ZERO
           DO 40 I = 1,27
             DO 50 J = 1,27
               MATA(I,J) = 0.0D0
               DQ(I) = 0.0D0
               MATC(I) = 0.0D0
 50          CONTINUE
 40        CONTINUE
           DO 60 I = 1,27
             MATB(I) = 0.0D0
 60        CONTINUE
*      INITIALIZE THE INITIAL VELOCITIES AND TRANSFORMATION MATRICIES
           DO 63 I = 1,3
             DO 64 J = 1,3
               RATE1(I)    = 0.0D0
               RATE2(I)    = 0.0D0
               RATE3(I)    = 0.0D0
               BRATE1(I)   = 0.0D0
               BRATE2(I)   = 0.0D0
               BRATE3(I)   = 0.0D0
               MAT1T (I,J) = 0.0D0
               MAT2T (I,J) = 0.0D0
               MAT3T (I,J) = 0.0D0
               MAT1R (I,J) = 0.0D0
               MAT2R (I,J) = 0.0D0
               MAT3R (I,J) = 0.0D0
 64          CONTINUE
 63        CONTINUE

DERIVATIVE
NOSORT
          CALL ERRSET (208,256,-1,1,1)
          LEVELQ = 0
          CALL UERSET(LEVELQ,LEVLDQ)
```

```
*       INITIALIZE MATRIX A AND B TO ZERO
              DO 70 I = 1,27
                DO 80 J = 1,27
                  MATA(I,J) = 0.0D0
 80           CONTINUE
 70         CONTINUE
              DO 90 I = 1,27
                MATB(I) = 0.0D0
                DQ(I) = 0.0D0
 90         CONTINUE

*       INPUT JOINT EQUATIONS
*       JOINT ZERO  EQUATIONS
*       AB = AG1 + (WD1 X RB/G1) + W1 X (W1 X RB/G1)
                 VECTA0(1) = WDX(1)
                 VECTA0(2) = WDY(1)
                 VECTA0(3) = WDZ(1)
                 RBG1(1) = JX0 - LCOGX(1)
                 RBG1(2) = JY0 - LCOGY(1)
                 RBG1(3) = JZ0 - LCOGZ(1)
               CALL CPROD(VECTA0,RBG1,MIA0,MJA0,MKA0)
                 VECTA0(1) = W1(1)
                 VECTA0(2) = W1(2)
                 VECTA0(3) = W1(3)
               CALL CPROD(VECTA0,RBG1,MIB0,MJB0,MKB0)
                 VECTB0(1) = MIB0
                 VECTB0(2) = MJB0
                 VECTB0(3) = MKB0
               CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)
*       JOINT ONE EQUATIONS---
*       AA = AG1 + (WD1 X RA/G1) + W1 X (W1 X RA/G1)
                 VECTA1(1) = WDX(1)
                 VECTA1(2) = WDY(1)
                 VECTA1(3) = WDZ(1)
                 RAG1(1) = JX1 - LCOGX(1)
                 RAG1(2) = JY1 - LCOGY(1)
                 RAG1(3) = JZ1 - LCOGZ(1)
               CALL CPROD(VECTA1,RAG1,MIA1,MJA1,MKA1)
                 VECTA1(1) = W1(1)
                 VECTA1(2) = W1(2)
                 VECTA1(3) = W1(3)
               CALL CPROD (VECTA1,RAG1,MIB1,MJB1,MKB1)
                 VECTB1(1) = MIB1
                 VECTB1(2) = MJB1
                 VECTB1(3) = MKB1
               CALL CPROD (VECTA1,VECTB1,MIC1,MJC1,MKC1)
*        AB = AG2 + (WD2 X RB/G2) + W2 X (W2 X RB/G2)
                 VECTA1(1) = WDX(2)
                 VECTA1(2) = WDY(2)
                 VECTA1(3) = WDZ(2)
                 RBG2(1) = JX1 - LCOGX(2)
                 RBG2(2) = JY1 - LCOGY(2)
                 RBG2(3) = JZ1 - LCOGZ(2)
              CALL CPROD (VECTA1,RBG2,MIA2,MJA2,MKA2)
                 VECTA1(1) = W2(1)
                 VECTA1(2) = W2(2)
```

```
                VECTA1(3) = W2(3)
              CALL CPROD (VECTA1,RBG2,MIB2,MJB2,MKB2)
                VECTB1(1) = MIB2
                VECTB1(2) = MJB2
                VECTB1(3) = MKB2
              CALL CPROD (VECTA1,VECTB1,MIC2,MJC2,MKC2)

*       JOINT TWO EQUATIONS
*       AA = AG2 + (WD2 X RA/G2) + W2 X (W2 X RA/G2)
                VECTA2(1) = WDX(2)
                VECTA2(2) = WDY(2)
                VECTA2(3) = WDZ(2)
                RAG2(1) = JX2 - LCOGX(2)
                RAG2(2) = JY2 - LCOGY(2)
                RAG2(3) = JZ2 - LCOGZ(2)
              CALL CPROD (VECTA2,RAG2,MIA3,MJA3,MKA3)
                VECTA2(1) = W2(1)
                VECTA2(2) = W2(2)
                VECTA2(3) = W2(3)
              CALL CPROD (VECTA2,RAG2,MIB3,MJB3,MKB3)
                VECTB2(1) = MIB3
                VECTB2(2) = MJB3
                VECTB2(3) = MKB3
              CALL CPROD(VECTA2,VECTB2,MIC3,MJC3,MKC3)
*       AB = AG3 + (WD3 X RB/G3) + W3 X (W3 X RB/G3)
                VECTA2(1) = WDX(3)
                VECTA2(2) = WDY(3)
                VECTA2(3) = WDZ(3)
                RBG3(1) = JX2 - LCOGX(3)
                RBG3(2) = JY2 - LCOGY(3)
                RBG3(3) = JZ2 - LCOGZ(3)
              CALL CPROD (VECTA2,RBG3,MIA4,MKA4,MKA4)
                VECTA2(1) = W3(1)
                VECTA2(2) = W3(2)
                VECTA2(3) = W3(3)
              CALL CPROD (VECTA2,RBG3,MIB4,MJB4,MKB4)
                VECTB2(1) = MIB4
                VECTB2(2) = MJB4
                VECTB2(3) = MKB4
              CALL CPROD (VECTA2,VECTB2,MIC4,MJC4,MKC4)
*       SUM OF MOMENTS EQUATIONS
         DO 100 I = 1,3
*    COMPUTE HX,HDOT X,HY,HDOT Y, HZ,HDOT Z
                RX(I,1) = -L(I,1) * DCOS(DRCRAX(I))
                RX(I,2) =  L(I,2) * DCOS(DRCRAX(I))
                RY(I,1) = -L(I,1) * DCOS(DRCRAY(I))
                RY(I,2) =  L(I,2) * DCOS(DRCRAY(I))
                RZ(I,1) = -L(I,1) * DCOS(DRCRAZ(I))
                RZ(I,2) =  L(I,2) * DCOS(DRCRAZ(I))
        IXX(I,1) = MASS(I,1) * ((RY(I,1) * RY(I,1)) + (RZ(I,1) * RZ(I,1)))
        IXX(I,2) = MASS(I,2) * ((RY(I,2) * RY(I,2)) + (RZ(I,2) * RZ(I,2)))
        IXXT(I)  = IXX(I,1) + IXX(I,2)
        IF (IXXT(I) .LE. .020) THEN
        IXXT(I)  = .020
        ELSE
        IXXT(I) = IXXT(I)
        END IF
```

```
      IXY(I,1) = MASS(I,1) * RX(I,1) * RY(I,1)
      IXY(I,2) = MASS(I,2) * RX(I,2) * RY(I,2)
      IXYT(I)  = IXY(I,1) + IXY(I,2)

      IXZ(I,1) = MASS(I,1) * RZ(I,1) * RX(I,1)
      IXZ(I,2) = MASS(I,2) * RZ(I,2) * RX(I,2)
      IXZT(I)  = IXZ(I,1) + IXZ(I,2)

      HDX(1) = WDX(1) * IXX(I,1)- WDZ(I) * IXZ(I,1) - WDY(I) * IXY(I,1)
      HDX(2) = WDX(2) * IXX(I,2)- WDZ(I) * IXZ(I,2) - WDY(I) * IXY(I,2)

      IYY(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RZ(I,1) * RZ(I,1)))
      IYY(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RZ(I,2) * RZ(I,2)))
      IYYT(I)  = IYY(I,1) + IYY(I,2)
      IF (IYYT(I) .LE. .020) THEN
      IYYT(I)  = .020
      ELSE
      IYYT(I)  = IYYT(I)
      END IF

      IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
      IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
      IYZT(I)  = IYZ(I,1) + IYZ(I,2)

      HDY(1) = WDY(I) * IYY(I,1) - WDX(I) * IXY(I,1) - WDZ(I) * IYZ(I,1)
      HDY(2) = WDY(I) * IYY(I,2) - WDX(I) * IXY(I,2) - WDZ(I) * IYZ(I,2)

      IZZ(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RY(I,1) * RY(I,1)))
      IZZ(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RY(I,2) * RY(I,2)))
      IZZT(I)  = IZZ(I,1) + IZZ(I,2)
      IF (IZZT(I) .LE. .020) THEN
      IZZT(I)  = .020
      ELSE
      IZZT(I)  = IZZT(I)
      END IF

      HDZ(1) = WDZ(I) * IZZ(I,1) - WDX(I) * IXZ(I,1) - WDY(I) * IYZ(I,1)
      HDZ(2) = WDZ(I) * IZZ(I,2) - WDX(I) * IXZ(I,2) - WDY(I) * IYZ(I,2)

            SUMHDX(I) = HDX(1) + HDX(2)
            SUMHDY(I) = HDY(1) + HDY(2)
            SUMHDZ(I) = HDZ(1) + HDZ(2)

 100    CONTINUE
*      ENTER CONSTANTS INTO MATRIX A
*      LINK ONE
*      SUM OF FORCES IN THE X DIRECTION
          MATA(1,1)  =  1.0D0
          MATA(1,4)  =  MASS1
          MATA(1,10) = -1.0D0
*      SUM OF FORCES IN Y DIRECTION
          MATA(2,2)  =  1.0D0
          MATA(2,5)  =  MASS1
          MATA(2,11) = -1.0D0
*      SUM OF FORCES IN Z DIRECTION
          MATA(3,3)  =  1.0D0
          MATA(3,6)  =  MASS1
          MATA(3,12) = -1.0D0
*      EQUATIONS AT JOINT ZERO
*      IN THE X DIRECTION
          MATA(4,4) =  1.0D0
          MATA(4,8) =  RBG1(3)
          MATA(4,9) = -RBG1(2)
*      IN THE Y DIRECTION
          MATA(5,5) =  1.0D0
          MATA(5,7) = -RBG1(3)
          MATA(5,9) =  RBG1(1)
```

```
*       IN THE Z DIRECTION
            MATA(6,6) =  1.0D0
            MATA(6,7) =  RBG1(2)
            MATA(6,8) = -RBG1(1)
*       SUM OF MOMENTS EQUATIONS FOR LINK ONE IN THE X,Y,Z DIRECTIONS
            MATA(7,2)  =  RBG1(3)
            MATA(7,3)  = -RBG1(2)
            MATA(7,7)  = -IXXT(1)
            MATA(7,8)  =  IXYT(1)
            MATA(7,9)  =  IXZT(1)
            MATA(7,11) = -RAG1(3)
            MATA(7,12) =  RAG1(2)
            MATA(8,1)  = -RBG1(3)
            MATA(8,3)  =  RBG1(1)
            MATA(8,7)  =  IXYT(1)
            MATA(8,8)  = -IYYT(1)
            MATA(8,9)  =  IYZT(1)
            MATA(8,10) =  RAG1(3)
            MATA(8,12) = -RAG1(1)
            MATA(9,1)  =  RBG1(2)
            MATA(9,2)  = -RBG1(1)
            MATA(9,7)  =  IXZT(1) + IXZT(2) + IXZT(3)
            MATA(9,8)  =  IYZT(1) + IYZT(2) + IYZT(3)
            MATA(9,9)  = -IZZT(1) - IZZT(2) - IZZT(3)
            MATA(9,10) = -RAG1(2)
            MATA(9,11) =  RAG1(1)
*       LINK TWO
*       SUM OF FORCES IN X DIRECTION
            MATA(10,10) =  1.0D0
            MATA(10,13) =  MASS2
            MATA(10,19) = -1.0D0
*       SUM OF FORCES IN THE Y DIRECTION
            MATA(11,11) =  1.0D0
            MATA(11,14) =  MASS2
            MATA(11,20) = -1.0D0
*       SUM OF FORCES IN THE Z DIRECTION
            MATA(12,12) =  1.0D0
            MATA(12,15) =  MASS2
            MATA(12,21) = -1.0D0
*       EQUATIONS AT JOINT ONE
*       IN THE X DIRECTION
            MATA(13,4)  = -1.0D0
            MATA(13,8)  = -RAG1(3)
            MATA(13,9)  =  RAG1(2)
            MATA(13,13) =  1.0D0
            MATA(13,17) =  RBG2(3)
            MATA(13,18) = -RBG2(2)
*       IN THE Y DIRECTION
            MATA(14,5)  = -1.0D0
            MATA(14,7)  =  RAG1(3)
            MATA(14,9)  = -RAG1(1)
            MATA(14,14) =  1.0D0
            MATA(14,16) = -RBG2(3)
            MATA(14,18) =  RBG2(1)
*       IN THE Z DIRECTION
            MATA(15,6)  = -1.0D0
            MATA(15,7)  = -RAG1(2)
            MATA(15,8)  =  RAG1(1)
            MATA(15,15) =  1.0D0
            MATA(15,16) =  RBG2(2)
```

```
            MATA(15,17) = -RBG2(1)
*       SUM OF MOMENTS EQUATIONS FOR LINK TWO IN THE X,Y,Z DIRECTIONS
            MATA(16,11) =  RBG2(3)
            MATA(16,12) = -RBG2(2)
            MATA(16,16) = -IXXT(2)
            MATA(16,17) =  IXYT(2)
            MATA(16,18) =  IXZT(2)
            MATA(16,20) = -RAG2(3)
            MATA(16,21) =  RAG2(2)

            MATA(17,10) = -RBG2(3)
            MATA(17,12) =  RBG2(1)
            MATA(17,16) =  IXYT(2)
            MATA(17,17) = -IYYT(2)
            MATA(17,18) =  IYZT(2)
            MATA(17,19) =  RAG2(3)
            MATA(17,21) = -RAG2(1)

            MATA(18,10) =  RBG2(2)
            MATA(18,11) = -RBG2(1)
            MATA(18,16) =  IXZT(2) + IXZT(3)
            MATA(18,17) =  IYZT(2) + IYZT(3)
            MATA(18,18) = -IZZT(2) - IZZT(3)
            MATA(18,19) = -RAG2(2)
            MATA(18,20) =  RAG2(1)

*       LINK THREE
*       SUM OF FORCES IN THE X DIRECTION
            MATA(19,19) =  1.0D0
            MATA(19,22) =  MASS3
*       SUM OF FORCES IN THE Y DIRECTION
            MATA(20,20) =  1.0D0
            MATA(20,23) =  MASS3
*       SUM OF FORCES IN THE Z DIRECTION
            MATA(21,21) =  1.0D0
            MATA(21,24) =  MASS3
*       EQUATIONS AT JOINT TWO
*       IN THE X DIRECTION
            MATA(22,13) = -1.0D0
            MATA(22,17) = -RAG2(3)
            MATA(22,18) =  RAG2(2)
            MATA(22,22) =  1.0D0
            MATA(22,26) =  RBG3(3)
            MATA(22,27) = -RBG3(2)
*       IN THE Y DIRECTION
            MATA(23,14) = -1.0D0
            MATA(23,16) =  RAG2(3)
            MATA(23,18) = -RAG2(1)
            MATA(23,23) =  1.0D0
            MATA(23,25) = -RBG3(3)
            MATA(23,27) =  RBG3(1)
*       IN THE Z DIRECTION
            MATA(24,15) = -1.0D0
            MATA(24,16) = -RAG2(2)
            MATA(24,17) =  RAG2(1)
            MATA(24,24) =  1.0D0
            MATA(24,25) =  RBG3(2)
            MATA(24,26) = -RBG3(1)
*       SUM OF MOMENTS EQUATIONS FOR LINK THREE IN THE X,Y,Z DIRECTIONS
            MATA(25,20) =  RBG3(3)
            MATA(25,21) = -RBG3(2)
            MATA(25,25) = -IXXT(3)
```

```
            MATA(25,26) =  IXYT(3)
            MATA(25,27) =  IXZT(3)
            MATA(26,19) = -RBG3(3)
            MATA(26,21) =  RBG3(1)
            MATA(26,25) =  IXYT(3)
            MATA(26,26) = -IYYT(3)
            MATA(26,27) =  IYZT(3)
            MATA(27,19) =  RBG3(2)
            MATA(27,20) = -RBG3(1)
            MATA(27,25) =  IXZT(3)
            MATA(27,26) =  IYZT(3)
            MATA(27,27) = -IZZT(3)
*         GO TO 1112
*       INITIALIZE MATRIX ACCORDING TO CONSTRAINT
*       CONSTRAINTS GROUP 1 WHEN ONLY LINK THREE IS IN MOTION
*         DO 118 I = 1,18
*           DO 18 J = 1,27
*             MATA(I,J) = 0.0
*             MATA(I,I) = 1.0
*             MATB(I)   = 0.0
*18         CONTINUE
*118       CONTINUE
*
*         DO 181 I = 19,27
*           DO 81 J = 1,18
*             MATA(I,J) = 0.0
*81         CONTINUE
*181         CONTINUE
*         GO TO 1111
*       CONSTRAINTS GROUP 2 WHEN LINK TWO AND THREE ARE IN MOTION
*         DO 19 I = 1,9
*           DO 191 J = 1,27
*             MATA(I,J) = 0.0D0
*             MATA(I,I) = 1.0 D0
*             MATB(I)   = 0.0D0
*
*             MATA(17,J) = 0.0D0
*             MATA(18,J) = 0.0D0
*             MATB(17)   = 0.0D0
*             MATB(18)   = 0.D0
*
*             MATA(J,17) = 0.0D0
*             MATA(J,18) = 0.0D0
*
*             MATA(17,17) = 1.0D0
*             MATA(18,18) = 1.0D0
*
*191         CONTINUE
*19        CONTINUE
*
*         DO 91 I = 10,27
*           DO 92 J = 1,9
*             MATA(I,J) = 0.0
*92         CONTINUE
*91        CONTINUE
*         GO TO 1111
*       CONSTRAINTS GROUP 3 WHEN THREE OF THE LINKS ARE IN MOTION
*         DO 78 J = 1,27
*           MATA(7,J)  = 0.0D0
*           MATA(8,J)  = 0.0D0
*           MATA(J,7)  = 0.0D0
*           MATA(J,8)  = 0.0D0
*           MATB(7)    = 0.0D0
*           MATB(8)    = 0.0D0
```

```
*          MATA(17,J) = 0.0D0
*          MATA(18,J) = 0.0D0
*          MATA(J,17) = 0.0D0
*          MATA(J,18) = 0.0D0
*          MATB(17)   = 0.0D0
*          MATB(18)   = 0.0D0
*          MATA(26,J) = 0.0D0
*          MATA(27,J) = 0.0D0
*          MATA(J,26) = 0.0D0
*          MATA(J,27) = 0.0D0
*          MATB(26)   = 0.0D0
*          MATB(27)   = 0.0D0
*          MATA(7,7)   = 1.0D0
*          MATA(8,8)   = 1.0D0
*          MATA(17,17) = 1.0D0
*          MATA(18,18) = 1.0D0
*          MATA(26,26) = 1.0D0
*          MATA(27,27) = 1.0D0
*78       CONTINUE
*   CONSTRAINT GROUP 4 THE FIRST LINK IS CONSTRAINED TO ROTATE IN Z DIR.
 1112     DO 48 I = 4,8
             DO 481 J = 1,27
                MATA(I,J) = 0.0D0
                MATA(I,I) = 1.0 D0
                MATB(I)   = 0.0D0
 481      CONTINUE
 48       CONTINUE
           DO 84 I = 9,27
              DO 841 J = 4,8
                 MATA(I,J) = 0.0
 841         CONTINUE
 84       CONTINUE
*   USE THE FOLLOWING SET OF INFORMATION WHEN THE ANGULAR VELOCITY IS
*   IN X DIRECTION REGARDLESS OF THE INITIAL CONFIGURATION
*   ENTER THE THEORITICAL VALUES ASSUMING THE LINK TWO AND THREE ARE
*   IN PLANAR MOTION AND ANGULAR VELOCITY IS IN THE X DIRECTION
*      LINK TWO
*      THEORITICAL ANGULAR VELOCITIES (APPLIED IN THE X DIRECTION)
           MATB(18) = 0.0D0
           MATB(17) = 0.0D0
           MATB(16) = -((PI**3) / 8.0D0) * DSIN(W * TIME)
           THDDOT   = MATB(16)
           THTDOT   = INTGRL((PI**2)/4.,THDDOT)
           THETRB   = INTGRL(0.,THTDOT)
           THETAB   = THETRB * RADEG
*      LINEAR VELOCITIES
           MATB(15) = -(THDDOT * RBG2(2)) + (THTDOT ** 2) * RBG2(3)
           MATB(14) =  (THDDOT * RBG2(3)) + (THTDOT ** 2) * RBG2(2)
           MATB(13) =  0.0D0
*      LINK THREE
*      ANGULAR VELOCITIES
           MATB(27) = 0.0D0
           MATB(26) = 0.0D0
           MATB(25) = 0.0D0
*
*      LINEAR VELOCITIES
           MATB(24) = MATB(15)+(THDDOT*RAG2(2))-(THTDOT**2)*(RAG2(3))
           MATB(23) = MATB(14)-(THDDOT*RAG2(3))-(THTDOT**2)*(RAG2(2))
           MATB(22) = MATB(13)
*   END OF THE INFORMATION FOR X DIRECTION
*
```

```
*   USE THE FOLLOWING SET OF INFORMATION WHEN THE ANGULAR VELOCITY IS
*   IN THE Y DIRECTION REGARDLESS OF THE INITIAL CONFIGURATION
*
*   ENTER THE THEORITICAL VALUES ASSUMING THE LINK TWO AND THREE ARE
*   IN PLANAR MOTION AND ANGULAR VELOCITY IS IN THE Y DIRECTION
*   LINK TWO
*   THEORITICAL ANGULAR VELOCITIES(APPLIED IN THE Y DIRECTION )
*
*        MATB(18) =  0.0D0
*        MATB(17) =  ((-PI**3)/8)*SIN(W*TIME)
*        MATB(16) =  0.0D0
*        THDDOT   =  MATB(17)
*        THTDOT   =  INTGRL((PI**2)/4.,THDDOT)
*        THETRB   =  INTGRL(0,INTGRL(PI**2/4.,THDDOT))
*        THETAB   =  THETRB * RADEG
*   LINEAR VELOCITIES
*        MATB(15) =  (THDDOT * RBG2(1)) + (THTDOT ** 2) * RBG2(3)
*        MATB(14) =  0.0D0
*        MATB(13) = -(THDDOT * RBG2(3)) + (THTDOT ** 2) * RBG2(1)
*   LINK THREE
*   ANGULAR VELOCITIES
*        MATB(27) = 0.0D0
*        MATB(26) = 0.0D0
*        MATB(25) = 0.0D0
*   LINEAR VELOCITIES
*        MATB(24) = MATB(15)-(THDDOT*RAG2(1))-(THTDOT**2)*(RAG2(3))
*        MATB(23) = MATB(14)
*        MATB(22) = MATB(13)+(THDDOT*RAG2(3))-(THTDOT**2)*(RAG2(1))
*  END OF THE INFORMATION FOR THE Y DIRECTION

*   USE THE FOLLOWING SET OF INFORMATION WHEN THE ANGULAR VELOCITY IS
*   IN THE Z DIRECTION REGARDLESS OF THE INITIAL CONFIGURATION
*   ENTER THE THEORITICAL VALUES ASSUMING THE LINK TWO AND THREE ARE
*   IN PLANAR MOTION AND ANGULAR VELOCITY IS IN THE Z DIRECTION
*   LINK TWO
*   THEORITICAL ANGULAR VELOCITIES(APPLIED IN THE Z DIRECTION )
*        MATB(16) = 0.0D0
*        MATB(17) = 0.0D0
*        MATB(18) = -((PI**3) / 8.0D0) * DSIN(W * TIME)
*        THDDOT   = MATB(18)
*        THTDOT   = INTGRL((PI**2)/4.,THDDOT)
*        THETRB   = INTGRL(0.,THTDOT)
*        THETAB   = THETRB * RADEG
*   LINEAR VELOCITIES
*        MATB(14) = -(THDDOT * RBG2(1)) + (THTDOT ** 2) * RBG2(2)
*        MATB(13) =  (THDDOT * RBG2(2)) + (THTDOT ** 2) * RBG2(1)
*        MATB(15) =  0.0D0
*   LINK THREE
*   ANGULAR VELOCITIES
*
*        MATB(27) = 0.0D0
*        MATB(26) = 0.0D0
*        MATB(25) = 0.0D0
*   LINEAR VELOCITIES
*        MATB(24) = MATB(15)
*        MATB(23) = MATB(14)+(THDDOT*RAG2(1))-(THTDOT**2)*(RAG2(2))
*        MATB(22) = MATB(13)-(THDDOT*RAG2(2))-(THTDOT**2)*(RAG2(1))
*
*   END OF THE INFORMATION FOR THE Z DIRECTION
*   NEXT SET OF STATEMENTS ARE COMMON IN ANY PLANAR MOTION AND THEY ARE
*   IN YHE CODE IN EVERY CASE. THESE TERMS ARE ACCELERATION OF THE LINK
```

```
*   ONE AND FORCES AT EACH JOINT
*  LINK ONE LINEAR AND ANGULAR ACCELERATIONS
           MATB(4) = 0.0D0
           MATB(5) = 0.0D0
           MATB(6) = 0.0D0
           MATB(7) = 0.0D0
           MATB(8) = 0.0D0
           MATB(9) = 0.0D0
*  FORCES
*   JOINT TWO
*
           MATB(21) = -MASS3 * MATB(24) - WG3
           MATB(20) = -MASS3 * MATB(23)
           MATB(19) = -MASS3 * MATB(22)
*      JOINT ONE
           MATB(12) = MATB(21) - MASS2 * MATB(15) -WG2
           MATB(11) = MATB(20) - MASS2 * MATB(14)
           MATB(10) = MATB(19) - MASS2 * MATB(13)
*      JOINT ZERO
           MATB(3)  = MATB(12) - MASS1 * MATB(6) -WG1
           MATB(2)  = MATB(11) - MASS1 * MATB(5)
           MATB(1)  = MATB(10) - MASS1 * MATB(4)
*   END OF THE INFORMATION
*   MULTIPLY MATA AND MATB
       DO 505 J = 1,27
           SUM1 = 0.0
              DO 555 K = 1,27
                 SUM1 = SUM1 + MATA(J,K) * MATB(K)
555          CONTINUE
           DQ(J) = SUM1
505    CONTINUE
        DO 506 I =1,27
           MATC(I) = DQ(I)
 506   CONTINUE
*      CALL EQUATION SOLVER PROGRAM FROM IMSL
        CALL LEQT2F(MATA,M,N,IA,DQ,IDGT,WKAREA,IER)
*      IF (IER .NE. 0) CALL ENDJOB
*      FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
*      LINK ONE
           AX1        = DQ(4)
           VELX1      = INTGRL(0.,AX1)
           LCOGX1     = INTGRL(X1,VELX1)
           LCOGX(1)   = LCOGX1
           AY1        = DQ(5)
           VELY1      = INTGRL(0.,AY1)
           LCOGY1     = INTGRL(Y1,VELY1)
           LCOGY(1)   = LCOGY1
           AZ1        = DQ(6)
           VELZ1      = INTGRL(0.,AZ1)
           LCOGZ1     = INTGRL(Z1,VELZ1)
           LCOGZ(1)   = LCOGZ1
           WD1X       = DQ(7)
           W1X        = INTGRL(0.,WD1X)
           WDX(1)     = WD1X
           W1(1)      = W1X
           WD1Y       = DQ(8)
           W1Y        = INTGRL(0.,WD1Y)
           WDY(1)     = WD1Y
           W1(2)      = W1Y
```

```
          WD1Z      = DQ(9)
          W1Z       = INTGRL(0.,WD1Z)
          WDZ(1)    = WD1Z
          W1(3)     = W1Z
*     TRANSFORMATION MATRIX FROM EARTH FIXED TO BODY FIXED COORDINATE
*     SYSTEM FOR THE LINK ONE
          MAT1R(1,1) = DCOS(RLRZ1) * DCOS(PTRY1)
          MAT1R(2,1) = DCOS(RLRZ1) * DSIN(PTRY1) * DSIN(YWRX1) -...
          DSIN(RLRZ1) * DCOS(YWRX1)
          MAT1R(3,1) = DCOS(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) +...
          DSIN(RLRZ1) * DSIN(YWRX1)
          MAT1R(1,2) = DSIN(RLRZ1) * DCOS(PTRY1)
          MAT1R(2,2) = DSIN(RLRZ1) * DSIN(PTRY1) * DSIN(YWRX1) +...
          DCOS(RLRZ1) * DCOS(YWRX1)
          MAT1R(3,2) = DSIN(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) -...
          DCOS(RLRZ1) * DSIN(YWRX1)
          MAT1R(1,3) = -DSIN(PTRY1)
          MAT1R(2,3) = DCOS(PTRY1) * DSIN(YWRX1)
          MAT1R(3,3) = DCOS(PTRY1) * DCOS(YWRX1)
*      GET THE VELOCITIES FOR LINK 1 IN BODY FIXED COOR. SYSTEM
          DO 605 J = 1,3
             SUM1 = 0.0D0
                DO 606 K = 1,3
                   SUM1 = SUM1 + MAT1R(J,K) * W1(K)
 606            CONTINUE
                BRATE1(J) = SUM1
 605       CONTINUE
*      TRANSFORMATION MATRIX FROM BODY FIXED TO EULER COORDINATE
*      SYSTEM FOR THE LINK ONE
          MAT1T(1,1) = 0.0D0
          MAT1T(2,1) = 1.0D0
          MAT1T(3,1) = 0.0D0
          MAT1T(1,2) =  DCOS(YWRX1)
          MAT1T(2,2) =  DTAN(PTRY1) * DSIN(YWRX1)
          MAT1T(3,2) =  1.0D0/DCOS(PTRY1) * DSIN(YWRX1)
          MAT1T(1,3) = -DSIN(YWRX1)
          MAT1T(2,3) =  DTAN(PTRY1) * DCOS(YWRX1)
          MAT1T(3,3) =  1.D0/DCOS(PTRY1) * DCOS(YWRX1)
*       GET THE YAW,PITCH AND THE ROLL RATES FOR LINK ONE
          DO 705 J = 1,3
             SUM1 = 0.0D0
                DO 706 K = 1,3
                   SUM1 = SUM1 + MAT1T(J,K) * BRATE1(K)
 706            CONTINUE
                RATE1(J) = SUM1
 705       CONTINUE
          RATE1X = RATE1(1)
          RATE1Y = RATE1(2)
          RATE1Z = RATE1(3)
          YWRX1 = INTGRL(0.,RATE1X)
          PTRY1 = INTGRL(0.,RATE1Y)
          RLRZ1 = INTGRL(0.,RATE1Z)
          YAWANX(1) = YWRX1 * RADEG
          PTCANY(1) = PTRY1 * RADEG
          ROLANZ(1) = RLRZ1 * RADEG
*       DIRECTION COSINES FOR THE FIRST LINK SAME FOR EACH CASE
          DRCSY(1) = DCOS(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) +...
          DSIN(RLRZ1) * DSIN(YWRX1)
```

```
          DRCSX(1) = DSIN(RLRZ1) * DSIN(PTRY1) * DCOS(YWRX1) -...
          DCOS(RLRZ1) * DSIN(YWRX1)
          DRCSZ(1) = DCOS(PTRY1) * DCOS(YWRX1)
*       GET THE ANGLES AS RADIANS
          DRCRAX(1) = DACOS(DRCSX(1))
          DRCRAY(1) = DACOS(DRCSY(1))
          DRCRAZ(1) = DACOS(DRCSZ(1))
*       CONVERT THE DIRECTION COSINES TO DEGREES
          DRCANX(1) = DACOS(DRCSX(1)) * RADEG
          DRCANY(1) = DACOS(DRCSY(1)) * RADEG
          DRCANZ(1) = DACOS(DRCSZ(1)) * RADEG
*        LINK TWO
9          AX2       = DQ(13)
             VELX2     = INTGRL(0.,AX2)
             LCOGX2    = INTGRL(X2,VELX2)
             LCOGX(2)  = LCOGX2
             AY2       = DQ(14)
             VELY2     = INTGRL(0.,AY2)
             LCOGY2    = INTGRL(Y2,VELY2)
             LCOGY(2)  = LCOGY2
             AZ2       = DQ(15)
             VELZ2     = INTGRL(0.,AZ2)
             LCOGZ2    = INTGRL(Z2,VELZ2)
             LCOGZ(2)  = LCOGZ2
             WD2X      = DQ(16)
             W2X       = INTGRL((PI**2)/4.,WD2X)
*       USE THE INIT. COND. WITH ONLY WHICHEVER VELOCITY APPLIED
*       AND KEEP THE TWO OTHER ANG. VEL. INIT. COND. AS ZERO
*       USE THE NEXT STATEMENT IF THE ANGULAR VELOCITY IS IN THE X DIR.
          THETRD    = INTGRL(0.,W2X)
          WDX(2)    = WD2X
          W2(1)     = W2X
          WD2Y      = DQ(17)
          W2Y       = INTGRL(0.,WD2Y)
*       USE THE NEXT STATEMENT IF THE ANGULAR VELOCITY IS IN THE Y DIR.
*          THETRD    = INTGRL(0.,W2Y)
          WDY(2)    = WD2Y
          W2(2)     = W2Y
          WD2Z      = DQ(18)
          W2Z       = INTGRL(0.,WD2Z)
*       USE THE NEXT STATEMENT IF THE ANGULAR VELOCITY IS IN THE Z DIR.
*         THETRD    = INTGRL(0.,W2Z)
          WDZ(2)    = WD2Z
          W2(3)     = W2Z
          THETAD  = THETRD * RADEG
          ERROR   = ABS(((THETAD-THETAB)/180.) * 100)
*       TRANSFORMATION MATRIX FROM EARTH FIXED TO BODY FIXED COORDINATE
*       SYSTEM FOR THE LINK TWO
          MAT2R(1,1) = DCOS(RLRZ2) * DCOS(PTRY2)
          MAT2R(2,1) = DCOS(RLRZ2) * DSIN(PTRY2) * DSIN(YWRX2) -...
          DSIN(RLRZ2) * DCOS(YWRX2)
          MAT2R(3,1) = DCOS(RLRZ2) * DSIN(PTRY2) * DCOS(YWRX2) +...
          DSIN(RLRZ2) * DSIN(YWRX2)
          MAT2R(1,2) = DSIN(RLRZ2) * DCOS(PTRY2)
          MAT2R(2,2) = DSIN(RLRZ2) * DSIN(PTRY2) * DSIN(YWRX2) +...
```

```
          DCOS(RLRZ2) * DCOS(YWRX2)
          MAT2R(3,2) = DSIN(RLRZ2) * DSIN(PTRY2) * DCOS(YWRX2) -...
          DCOS(RLRZ2) * DSIN(YWRX2)
          MAT2R(1,3) = -DSIN(PTRY2)
          MAT2R(2,3) = DCOS(PTRY2) * DSIN(YWRX2)
          MAT2R(3,3) = DCOS(PTRY2) * DCOS(YWRX2)
*       GET THE VELOCITIES FOR LINK 2 IN BODY FIXED COOR. SYSTEM
          DO 607 J = 1,3
            SUM1 = 0.0D0
              DO 608 K = 1,3
                SUM1 = SUM1 + MAT2R(J,K) * W2(K)
 608          CONTINUE
              BRATE2(J) = SUM1
 607      CONTINUE
*       TRANSFORMATION MATRIX FROM BODY FIXED TO EULER COOR. SYSTEM
*       FOR THE LINK TWO
          MAT2T(1,1) = 0.0D0
          MAT2T(2,1) = 1.0D0
          MAT2T(3,1) = 0.0D0
          MAT2T(1,2) =  DCOS(YWRX2)
          MAT2T(2,2) =  DTAN(PTRY2) * DSIN(YWRX2)
          MAT2T(3,2) =  1.0D0/DCOS(PTRY2) * DSIN(YWRX2)
          MAT2T(1,3) = -DSIN(YWRX2)
          MAT2T(2,3) =  DTAN(PTRY2) * DCOS(YWRX2)
          MAT2T(3,3) =  1.0D0/DCOS(PTRY2) * DCOS(YWRX2)
*       GET THE YAW,PITCH AND THE ROLL RATES FOR LINK TWO
          DO 707 J = 1,3
            SUM1 = 0.0D0
              DO 708 K = 1,3
                SUM1 = SUM1 + MAT2T(J,K) * BRATE2(K)
 708          CONTINUE
              RATE2(J) = SUM1
 707      CONTINUE
          RATE2X = RATE2(1)
          RATE2Y = RATE2(2)
          RATE2Z = RATE2(3)
*       USE THE NEXT THREE STATEMENTS FOR CASE A
          YWRX2 = INTGRL(0.,RATE2X)
          PTRY2 = INTGRL(0.,RATE2Y)
          RLRZ2 = INTGRL(-PI/2.,RATE2Z)
*       USE THE NEXT THREE STATEMENTS FOR CASE B
*         YWRX2 = INTGRL(0.,RATE2X)
*         PTRY2 = INTGRL(0.,RATE2Y)
*         RLRZ2 = INTGRL(PI/2.,RATE2Z)
*       USE THE NEXT THREE STATEMENTS FOR CASE C
*         YWRX2 = INTGRL(0.,RATE2X)
*         PTRY2 = INTGRL(0.,RATE2Y)
*         RLRZ2 = INTGRL(0.,RATE2Z)
            RATE2(1) = RATE2X
            RATE2(2) = RATE2Y
            RATE2(3) = RATE2Z
          YAWANX(2) = YWRX2 * RADEG
          PTCANY(2) = PTRY2 * RADEG
          ROLANZ(2) = RLRZ2 * RADEG
*       USE THE NEXT SET OF THE DIRECTION COSINES FOR LINK TWO FOR CASE A
          DRCSY(2) = DCOS(RLRZ2) * DSIN(PTRY2) * DSIN(YWRX2) -...
          DSIN(RLRZ2) * DCOS(YWRX2)
          DRCSX(2) = DSIN(RLRZ2) * DSIN(PTRY2) * DSIN(YWRX2) +...
```

```
          DCOS(RLRZ2) * DCOS(YWRX2)
          DRCSZ(2) = DCOS(PTRY2) * DSIN(YWRX2)
*       USE THE NEXT SET OF THE DIRECTION COSINES FOR LINK TWO FOR CASE B
*
*         DRCSY(2) = DCOS(RLRZ2) * DCOS(PTRY2)
*         DRCSX(2) = DSIN(RLRZ2)*DCOS(PTRY2)
*
*         DRCSZ(2) = -DSIN(PTRY2)
*       USE THE NEXT SET OF THE DIRECTION COSINES FOR LINK TWO FOR CASE C
*         DRCSY(2) = DCOS(RLRZ2) * DSIN(PTRY2) * DCOS(YWRX2) +...
*         DSIN(RLRZ2) * DSIN(YWRX2)
*
*         DRCSX(2) = DSIN(RLRZ2) * DSIN(PTRY2) * DCOS(YWRX2) -...
*         DCOS(RLRZ2) * DSIN(YWRX2)
*         DRCSZ(2) = DCOS(PTRY2) * DCOS(YWRX2)
*       GET THE ANGLES AS RADIANS
          DRCRAX(2) = DACOS(DRCSX(2))
          DRCRAY(2) = DACOS(DRCSY(2))
          DRCRAZ(2) = DACOS(DRCSZ(2))
*       CONVERT THE DIRECTION COSINES TO DEGREES
          DRCANX(2) = DACOS(DRCSX(2)) * RADEG
          DRCANY(2) = DACOS(DRCSY(2)) * RADEG
          DRCANZ(2) = DACOS(DRCSZ(2)) * RADEG
*       FIND THE JOINT LOCATION
          JX1  =  (L(1,1) + L(1,2)) * DCOS(DRCRAX(1))
          JY1  =  (L(1,1) + L(1,2)) * DCOS(DRCRAY(1))
          JZ1  =  (L(1,1) + L(1,2)) * DCOS(DRCRAZ(1))
*       LINK THREE
6         AX3       = DQ(22)
            VELX3     = INTGRL(0.,AX3)
            LCOGX3    = INTGRL(X3,VELX3)
            LCOGX(3)  = LCOGX3
            AY3       = DQ(23)
            VELY3     = INTGRL(0.,AY3)
            LCOGY3    = INTGRL(Y3,VELY3)
            LCOGY(3)  = LCOGY3
            AZ3       = DQ(24)
            VELZ3     = INTGRL(0.,AZ3)
            LCOGZ3    = INTGRL(Z3,VELZ3)
            LCOGZ(3)  = LCOGZ3
            WD3X      = DQ(25)
            W3X       = INTGRL(0.,WD3X)
            WDX(3)    = WD3X
            W3(1)     = W3X
            WD3Y      = DQ(26)
            W3Y       = INTGRL(0.,WD3Y)
            WDY(3)    = WD3Y
            W3(2)     = W3Y
            WD3Z      = DQ(27)
            W3Z       = INTGRL(0.,WD3Z)
            WDZ(3)    = WD3Z
            W3(3)     = W3Z
*       TRANSFORMATION MATRIX FROM EARTH FIXED TO BODY FIXED COOR. SYSTEM
*       FOR THE LINK THREE
          MAT3R(1,1) = DCOS(RLRZ3) * DCOS(PTRY3)
          MAT3R(2,1) = DCOS(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) -...
          DSIN(RLRZ3) * DCOS(YWRX3)
          MAT3R(3,1) = DCOS(RLRZ3) * DSIN(PTRY3) * DCOS(YWRX3) +...
```

```
          DSIN(RLRZ3) * DSIN(YWRX3)
          MAT3R(1,2) = DSIN(RLRZ3) * DCOS(PTRY3)
          MAT3R(2,2) = DSIN(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) +...
          DCOS(RLRZ3) * DCOS(YWRX3)
          MAT3R(3,2) = DSIN(RLRZ3) * DSIN(PTRY3) * DCOS(YWRX3) -...
          DCOS(RLRZ3) * DSIN(YWRX3)
          MAT3R(1,3) = -DSIN(PTRY3)
          MAT3R(2,3) = DCOS(PTRY3) * DSIN(YWRX3)
          MAT3R(3,3) = DCOS(PTRY3) * DCOS(YWRX3)
*      GET THE VELOCITIES FOR LINK 3 IN BODY FIXED COOR. SYSTEM
          DO 609 J = 1,3
            SUM1 = 0.0D0
              DO 610 K = 1,3
                SUM1 = SUM1 + MAT3R(J,K) * W3(K)
 610        CONTINUE
            BRATE3(J) = SUM1
 609      CONTINUE
*      TRANSFORMATION MATRIX FROM BODY FIXED TO EULER COOR. SYSTEM
*      FOR THE LINK THREE
          MAT3T(1,1) = 0.0D0
          MAT3T(2,1) = 1.0D0
          MAT3T(3,1) = 0.0D0
          MAT3T(1,2) =  DCOS(YWRX3)
          MAT3T(2,2) =  DTAN(PTRY3) * DSIN(YWRX3)
          MAT3T(3,2) =  1.0D0/DCOS(PTRY3) * DSIN(YWRX3)
          MAT3T(1,3) = -DSIN(YWRX3)
          MAT3T(2,3) =  DTAN(PTRY3) * DCOS(YWRX3)
          MAT3T(3,3) =  1.0D0/DCOS(PTRY3) * DCOS(YWRX3)
*     GET THE YAW,PITCH AND THE ROLL RATES FOR LINK THREE
          DO 709 J = 1,3
            SUM1 = 0.0D0
              DO 710 K = 1,3
                SUM1 = SUM1 + MAT3T(J,K) * BRATE3(K)
 710        CONTINUE
            RATE3(J) = SUM1
 709      CONTINUE
          RATE3X = RATE3(1)
          RATE3Y = RATE3(2)
          RATE3Z = RATE3(3)
*      USE THE NEXT THREE FOR THE CASE A
          YWRX3 = INTGRL(0.,RATE3X)
          PTRY3 = INTGRL(0.,RATE3Y)
          RLRZ3 = INTGRL(-PI/2.,RATE3Z)
*      USE THE NEXT THREE FOR THE CASE B
*        YWRX3 = INTGRL(0.,RATE3X)
*        PTRY3 = INTGRL(0.,RATE3Y)
*        RLRZ3 = INTGRL(PI/2.,RATE3Z)
*      USE THE NEXT THREE FOR THE CASE C
*        YWRX3 = INTGRL(0.,RATE3X)
*        PTRY3 = INTGRL(0.,RATE3Y)
*        RLRZ3 = INTGRL(0.,RATE3Z)
          YAWANX(3) = YWRX3 * RADEG
          PTCANY(3) = PTRY3 * RADEG
          ROLANZ(3) = RLRZ3 * RADEG
*    USE THE NEXT SET OF THE DIRECTION COSINES FOR LINK THREE FOR CASE A
          DRCSY(3) = DCOS(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3) -...
          DSIN(RLRZ3) * DCOS(YWRX3)
```

```
            DRCSX(3) = DSIN(RLRZ3) * DSIN(PTRY3) * DSIN(YWRX3)+...
            DCOS(RLRZ3) * DCOS(YWRX3)
            DRCSZ(3) = DCOS(PTRY3) * DSIN(YWRX3)
*
*   USE THE NEXT SET OF THE DIRECTION COSINES FOR LINK THREE FOR CASE B
*        DRCSY(3) = DCOS(RLRZ3) * DCOS(PTRY3)
*        DRCSX(3) = DSIN(RLRZ3)*DCOS(PTRY3)
*
*        DRCSZ(3) = -DSIN(PTRY3)
*   USE THE NEXT SET OF THE DIRECTION COSINES FOR LINK THREE FOR CASE C
*        DRCSY(3) = DCOS(RLRZ3) * DSIN(PTRY3) * DCOS(YWRX3)+...
*        DSIN(RLRZ3) * DSIN(YWRX3)
*        DRCSX(3) = DSIN(RLRZ3) * DSIN(PTRY3) * DCOS(YWRX3)-...
*        DCOS(RLRZ3) * DSIN(YWRX3)
*        DRCSZ(3) = DCOS(PTRY3) * DCOS(YWRX3)
*      GET THE ANGLES AS RADIANS
            DRCRAX(3) = DACOS(DRCSX(3))
            DRCRAY(3) = DACOS(DRCSY(3))
            DRCRAZ(3) = DACOS(DRCSZ(3))
*      CONVERT THE DIRECTION COSINES TO DEGREES
            DRCANX(3) = DACOS(DRCSX(3)) * RADEG
            DRCANY(3) = DACOS(DRCSY(3)) * RADEG
            DRCANZ(3) = DACOS(DRCSZ(3)) * RADEG
*      FIND ANGLE BETWEEN LINK 2 AND LINK 3 TO CHECK IF THE ARM LINKS
*      are PASSING THROUGH THE SINGULAR POINTS
            ANG23X = DRCANX(2) - DRCANX(3)
            ANG23Y = DRCANY(2) - DRCANY(3)
            ANG23Z = DRCANZ(2) - DRCANZ(3)
            ANG12X = DRCANX(1) - DRCANX(2)
            ANG12Y = DRCANY(1) - DRCANY(2)
            ANG12Z = DRCANZ(1) - DRCANZ(2)
*      FIND THE JOINT LOCATION
            JX2 = JX1 + (L(2,1) + L(2,2)) * DCOS(DRCRAX(2))
            JY2 = JY1 + (L(2,1) + L(2,2)) * DCOS(DRCRAY(2))
            JZ2 = JZ1 + (L(2,1) + L(2,2)) * DCOS(DRCRAZ(2))
            TIPX = JX2 + (L(3,1) + L(3,2)) * DCOS(DRCRAX(3))
            TIPY = JY2 + (L(3,1) + L(3,2)) * DCOS(DRCRAY(3))
            TIPZ = JZ2 + (L(3,1) + L(3,2)) * DCOS(DRCRAZ(3))

END
STOP
FORTRAN
*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS
            SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
                 IMPLICIT REAL*8 (A-Z)
                 DIMENSION VECTA(3),VECTB(3)
                 MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
                 MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
                 MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
             RETURN
             END
```

LIST OF REFERENCES

1. Fu, K. S., Gonzales, R. C., Lee, C. S. G., Robotics: Control, Sensing, Vision, and Intelligence, pp. preface 9, McGraw-Hill Book Co., 1987.
2. Fu, K. S., Gonzales, R. C., Lee, C. S. G., Robotics: Control, Sensing, Vision, and Intelligence, pp. 36-41, McGraw-Hill Book Co., 1987.
3. Craig, J. J., Introduction to Robotic Mechanism and Control, pp. 146-149, Addison-Wesley Co., 1986.
4. Walker, M. W., and Orin, D. E., "Efficient Dynamic Computer Simulation of Robot Mechanism," ASME J. of Dynamics Systems, Measurements and Control Vol. 104, pp. 205-211.
5. Luh, J. Y. S., Walker, M.W., and Paul, R. P. C., "On-line Computational Scheme for Mechanical Manipulators," ASME J. of Dynamic Systems, Measurements and Control Vol. 102, pp. 69-76, June 1980.
6. Hollerbach, J. M., "A Recursive Langrangian Formulation of Manipulative Dynamics and a Comperative Study of Dynamics Formulation Complexity," IEEE Trans. SYST Man. and Cybern. Vol. SMC10 No. 11, pp. 730-736, 1980.
7. Paul. R., "The Mathematics of Computer Controlled Manipulators," Proceedings Joint Control Conference Vol. I, pp. 124-131, 1977.

8. Williams, R. J., and Seireg, A., "Interactive Modelling and Analysis of Open and Close Loop Dynamics System with Redundant Actuators," ASME J. of Mechanical Design Vol. 101 No. 3, pp. 407-416, July 1979.

9. Kane, T., and Levinson, D. A., "The Use of Kane's Dynamical Equations in Robotics," International J. of Robotics Research Vol. 2 No. 3, pp. 3-21, Fall 1983.

10. McCarthy, W. F., "Simulation of High Speed Motion of Rigid Revolute Mechanism," M. S. Thesis, Naval Postgraduate School, Monterey, California, December 1985.

11. Paul, R. P., and Stevenson, C. N., "Kinematics of Robot Wrists," International J. of Robotics Research Vol. 2 No. 1, pp. 31-38, Spring 1983.

12. Barker, K. L., "Kinematic Equations for Resolved Rate Control of an Industrial Robot Arm," NASA Technical Memo., 85685, pp. 1-35, November 1983.

13. Barker, K. L., Houck, J. A., and Carzoo, S. W., "Translational Control of a Graphicially Simulated Robot Arm by Kinematic Rate Equations that Overcome Joint Elbow Singularity," NASA Tech. Paper, 2376, December 1984.

14. Waldron, K. J., Wang, S., and Bolin, S. J., "A Study of the Jacobian Matrix of Serial Manipulators," ASME Design Engineering Technical Conference, Cambridge, Massachusetts, October 7-10, 1984.

15. Yoshikowa, T., "Analysis and Control of Robot Manipulators with Redundancy," in Robotics Research First International Symposium, ed. M. Brady and R. Paul, Cambridge, Massachusetts, pp. 735-748, 1984.

16. Hollerbach, J. M., "Optimum Kinematics Design for a Seven Degree of Freedom Manipulator," 2nd International Symposium of Robotics Research, Kyoto, Japan, pp. 349-356 August 20-23, 1984.

17. Luh, J. Y. S., and Gu, Y. L., "Industrial Robots with Seven Joints," IEEE International Conference on Robotics and Automation, pp. 1010-1015, March 25-28, 1985.

18. Khayyam, M., "Non-Singular Modelling of Rigid Manipulators," M. S. Thesis, Naval Postgraduate School, Monterey, California, December 1986.

19. Meriam, J. L., Engineering Mechanics Vol. 2, Dynamics, John Wiley and Sons, 1978.

20. Frank, A. A., McGhee, R. B., "Some Considerations Relating to the Design of Autopilots for Legged Vehicles," J. of Terramechanics Vol.6, pp. 23-35, 1969.

21. Torby, B. J., Advanced Dynamics for Engineers, pp. 218-221, HRW Series in Mechanical Engineering, 1984.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defence Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor D. L. Smith, Code 69Sm Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	10
5.	Professor Nunn, Code 69 Nn Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6.	Professor Chang, Code 69 Ck Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
7.	Ms. Mary Lacey Naval Surface Weapons Center White Oack, Code R402 Robotics Research and Development Center 10908 New Hampshire Avenue Silver Springs, Maryland 20903-5000	2
8.	Kara Kuvvetleri Komutanligi Arastirma Gelistirme Dairesi Bakanliklar, Ankara, Turkey	2
9.	Kara Harp Okulu Komutanligi Ogretim Kurulu Baskanligi Bakanliklar, Ankara, Turkey	2
10.	Topcu ve Fuze Okul Komutanligi Ogretim Kurulu Baskanligi Polatli, Ankara, Turkey	2

11. 1st Lieutenant S. Altinok, 2
Turkish Army
Eski Bagdat Caddesi No. 46/8
Kucukyali, Istanbul 81570, Turkey

Thesis
A4265
c.1

Altinok
A three dimensional non-singular modelling of rigid manipulators.